WEWoRC 2011

West European Workshop on Research in Cryptography

Chairs:
Stefan Lucks and Frederik Armknecht

Weimar, Germany, July 20-22, 2011

Conference Record

WEWoRC is jointly organized by

Bauhaus-Universität Weimar

Gold Sponsor

Silver Sponsor

Contents

We have listed all papers submitted and accepted for WEWoRC. Unfortunately, some authors had visa problems and hence could not attend the workshop. They have been listed in this conference record nevertheless. The responsibility for the correctness of the results presented lies solely with the corresponding authors. In particular, the papers collected in this conference record have not been peer reviewed.

RSA Vulnerabilities with Small Prime Difference

Marián Kühnel[1]

IT Security Group, RWTH Aachen, Germany

kuehnel@umic.rwth-aachen.de

Abstract. The security of the RSA cryptosystem is based on the assumption that recovering the private key from a public pair and factoring a modulus is a hard task. However, if the private key is smaller then some bound the system is considered to be insecure. An RSA modulus with small prime difference also significantly reduces the overall security. We show that the bound on small private key with respect to small prime difference can be further improved. Therefore, we adapt the technique of unravelled linearization for constructing lattices and although the adapted unravelled linearization is only a method for generating lattices in more elegant way, we yield a benefit compared to known bounds.

1 Introduction

The RSA cryptosystem is currently one of the most widely deployed asymmetric cryptosystems. From a mathematical point of view we generally try to break the RSA cryptosystem either by factorizing the modulus N or by exploiting dependencies in modular equations. In 1990, Wiener [5] demonstrated how one can reveal the private key from public pair if the original private key is smaller than $N^{\frac{1}{4}}$. Wiener also mentioned that his attack may sometimes work for private keys larger than $N^{\frac{1}{4}}$. This led to an intensive investigation of the modular equation used by private key generation. Boneh and Durfee improved Wiener's result and presented two approaches based on lattices which can find private keys smaller than $N^{0.292}$ in polynomial time [1]. These attacks were further extended by de Weger in case the modulus is a product of primes with small difference [6]. De Weger derived bounds for both variants where the upper bound for an extended Boneh-Durfee attack was not analyzed in general due to complicated restrictions in forming an adequate lattice. However, Herrman and May [3] introduced the technique of unravelled linearization[1] which performs a more suitable linearization on the modular equation and so exploits induced relations of the linearization itself. These relations are afterward used for the generation of a lattice. Although their technique did not exceed the bound given by Boneh and Durfee, they provide a new elegant solution to create an appropriate lattice without any complicated restrictions.

[1] originally introduced for exploiting output bits in power generators[2]

2 Short Preliminaries on Lattices

A lattice L is a set of all integer linear combinations of linearly independent vectors $u_1, u_2 \ldots, u_w \in \mathbb{Z}^n$ with $w \leq n$. One can also describe a lattice by its basis matrix B consisting of all the vectors $u_1, u_2 \ldots, u_w$ as row vectors. The dimension of a lattice $dim(L) = w$. If $w = n$, then the lattice is full rank and the absolute value of the determinant of a lattice basis matrix is equal to the determinant of a lattice. Since lattices obtained from unravelled linearization are always full rank and in addition triangular, the determinant is the product of the elements on the diagonal.

3 The Small Inverse Problem

Recall the RSA and the modulus N which is a product of two primes p and q. Then we denote the prime difference of p and q by $\Delta = |p - q|$. We can rewrite the small prime difference to $\Delta = N^\beta$ for $\beta = [\frac{1}{4}, \frac{1}{2}]$. In [6], de Weger pointed out that smaller β significantly improves the results of Wiener [5] and Boneh and Durfee [1]. Hence, in order to observe dependences between small prime difference and small private key, we define the private key in terms of N, concretely N^δ for $\delta \in [0, 1]$.

Lemma 1 *Let p and q are two primes of a modulus N and $N^\beta = \Delta = |p - q|$. Then $p + q \approx N^{2\beta - \frac{1}{2}}$.*

Proof: We sketch a proof similar to de Weger [6]. We have $\Delta^2 = (p+q)^2 - 4N = (p + q - 2\sqrt{N})(p + q + 2\sqrt{N})$. We know that $2\sqrt{N} < p + q < \frac{3}{\sqrt{2}}\sqrt{N}$. Hence

$$p + q - 2\sqrt{N} = \frac{\Delta^2}{p+q+2\sqrt{N}} < \frac{\Delta^2}{4\sqrt{N}}, \quad p + q + 2\sqrt{N} = \frac{\Delta^2}{p+q-2\sqrt{N}} > \frac{\Delta^2}{4\sqrt{N}}.$$

Therefore $p + q \approx \frac{\Delta^2}{4\sqrt{N}} \pm 2\sqrt{N} \approx N^{2\beta - \frac{1}{2}}$. □

Now we rewrite the common RSA modular equation into its equivalence and substitute terms $N + 1$ for A and $-(p + q)$ for y.

$$\begin{aligned} ed &= 1\ (mod\ \phi(N)) \\ ed &= 1 + x\phi(N) \\ ed &= 1 + x(N + 1 - (p + q)) \\ 0 &= 1 + x(A + y)\ (mod\ e). \end{aligned}$$

If modulus N has the same order of magnitude as public key e (i.g. $e \approx N^\alpha, \alpha \approx 1$) then we can solve the small inverse problem for a given polynomial $f(x, y) = 1 + x(A + y)\ (mod\ e)$ satisfying

$$f(x_0, y_0) \equiv 0\ (mod\ e)\ where\ |x_0| < N^\delta\ and |y_0| < N^{2/beta - \frac{1}{2}}.$$

The idea of solving the small inverse problem is to generate a set of coprime polynomials to the input polynomial $f(x, y)$ which contains the same roots over integers and then use basis reduction and root finding techniques to reveal exact roots.

4 The adapted unravelled linearization with Small Prime Difference

The underlying polynomial $f(x,y) = 1 + x(A+y)$ $(mod\ e)$ is the one used by Boneh and Durfee to generate a basis matrix. They also identified a sublattice $L^{'}$ which provides an improved bound on δ. However, for extracting the sublattice $L^{'}$ they introduced complicated geometrically progressive matrices [1] with a non triangular basis structure. The only approach which effectively reveals the sublattice $L^{'}$ and also keeps triangular structure needed for trivial determinant calculation is the unravelled linearization method. In this approach, Herrman and May [3] joined together the monomials of an underlaying bivariate polynomial

$$\underbrace{1+xy}_{u} + Ax\ mod\ e$$

and obtained a linear polynomial $\bar{f}(u,x) = u + Ax$ $(mod\ e)$ and additionally a relation $xy = u - 1$. Then in order to find small roots they fixed integers m and t, $t \leq m$ and constructed lattice from underlying polynomial $\bar{f}(u,x) = u + Ax$ using polynomials

$$\begin{aligned} &\bar{g}_{i,k} := x^i \bar{f}^k e^{m-k} \text{ for } k = 0,\dots,m \text{ and } i = 0,\dots,m-k \\ &\bar{h}_{j,k} := y^j \bar{f}^k e^{m-k} \text{ for } j = 1,\dots,t \text{ and } k = \left\lfloor \tfrac{m}{t} \right\rfloor j,\dots,m \end{aligned}$$

It is an important fact that $\tau = \frac{t}{m} \leq 1$. Otherwise, we would obtain a non triangular basis matrix with properties equivalent to de Weger's approach. Another crucial observation is that each generated row polynomial introduces only one new monomial and all other terms in each row are known from previous polynomials or can be substituted by the term $u-1$ obtained from the original linearization where we substituted $u = xy+1$. Therefore, the lattice is generated by a lower triangular basis matrix. An example of an basis matrix generated for parameters $m = 2$ and $t = 2$ from $\bar{g}_{i,k}$ and $\bar{h}_{j,k}$ is shown in Figure 1. The values X, Y and U denotes upper bounds of the solution for respective variable.

We mentioned that the basis matrix in unravelled linearization approach has always triangular basis matrix and hence entries on the diagonal indicate the determinant. Steps needed for derivating distinct contributions of the upper bounds to the determinant are explained in [3]. We give here only their results where the s_x denotes the contribution of the upper bound X to the determinant.

$$s_x = \sum_{k=0}^{m} \sum_{i=0}^{m-k} = \frac{1}{6}m^3 + o(m^3)$$

$$s_y = \sum_{j=1}^{\tau m} \sum_{k=\frac{1}{\tau}j}^{m} = \frac{\tau^2}{6}m^3 + o(m^3)$$

$$\begin{array}{c|ccccccccc}
 & 1 & x & u & x^2 & ux & u^2 & uy & u^2y & u^2y^2 \\
\hline
e^2 & e^2 & & & & & & & & \\
xe^2 & & e^2xX & & & & & & & \\
\bar{f}e & & eAxX & euU & & & & & & \\
x^2e^2 & & & & e^2x^2X^2 & & & & & \\
x\bar{f}e & & & & eAx^2X^2 & euUxX & & & & \\
\bar{f}^2 & & & & A^2x^2X^2 & 2AuUxX & u^2U^2 & & & \\
y\bar{f}e & -eA & & eAuU & & & & uUyY & & \\
y\bar{f}^2 & & -A^2xX & -2AuU & & A^2uUxX & 2Au^2U^2 & & u^2U^2yY & \\
y^2\bar{f}^2 & A^2 & & -2A^2uU & & & A^2u^2U^2 & -2AuUyY & 2Au^2U^2yY & u^2U^2y^2Y^2
\end{array}$$

Fig. 1. A lattice using unravelled linearization with parameters $m = 2$ and $t = 2$.

$$s_u = \sum_{k=0}^{m} \sum_{i=0}^{m-k} k + \sum_{j=1}^{\tau m} \sum_{k=\frac{1}{\tau}j}^{m} k = \left(\frac{1}{6} + \frac{\tau}{3}\right) m^3 + o(m^3)$$

$$s_e = \sum_{k=0}^{m} \sum_{i=0}^{m-k} (m-k) + \sum_{j=1}^{\tau m} \sum_{k=\frac{1}{\tau}j}^{m} (m-k) = \left(\frac{1}{3} + \frac{\tau}{6}\right) m^3 + o(m^3)$$

$$dim(L) = \sum_{k=0}^{m} \sum_{i=0}^{m-k} 1 + \sum_{j=1}^{\tau m} \sum_{k=\frac{1}{\tau}j}^{m} 1 = \left(\frac{1}{2} + \frac{\tau}{2}\right) m^2 + o(m^2)$$

Recall that our goal is to examine the impact of unravelled linearization on small difference of primes p and q. Therefore, we replace the parameters given in [3] for $X = N^{\delta}$, $Y = N^{2\beta-\frac{1}{2}}$ and $U = N^{\delta+2\beta-2}$. This leads to a more general determinant[2]

$$\begin{gathered}
det(L) = X^{s_x} Y^{s_y} U^{s_u} e^{s_e} \leq e^{m \cdot dim(L)} \\
N^{\delta s_x + (2\beta-\frac{1}{2})s_y + (\delta+2\beta-\frac{1}{2})s_u + s_e} \leq N^{dim(L)m} \\
N^{m^3(\frac{\delta}{6} + (2\beta-\frac{1}{2})\frac{\tau^2}{6} + (\delta+2\beta-\frac{1}{2})(\frac{1}{6}+\frac{\tau}{3}) + \frac{1}{3} + \frac{\tau}{6} - \frac{1}{2} - \frac{\tau}{2})} \leq 0
\end{gathered}$$

The left side is minimal for $\tau = \frac{3-2\delta-4\beta}{4\beta-1}$. Plugging the optimized τ back to the equation leads to $\delta = 1 - \sqrt{2\beta - \frac{1}{2}}$ which is asymptotically equal to the bound de Weger [6] achieved by expoiting complicated geometrically progressive matrices [1]. Unfortunately, these geometrically progressive matrices are strictly defined only for $\delta \leq 0.5$ [6]. Our unravelled linearization solution has also one restriction ($\tau \leq 1$) which is satisfied for values $\delta \leq 0.5$ and after reaching the point $\delta = 0.5$ we are "stopped" due to the inability to derive an adequate τ. However, if we continue with $\delta > 0.5$ for constant $\tau = 1$ and omit that the method won't be optimized any more, then we can obtain a benefit. The exact impact on boundary function is shown in Figure 2. The shaded area depicts concrete advantage compared to the (improved) Boneh-Durfee attack and de Weger's result and the green bold line shows the current boundary function on δ.

[2] for more information on determinant calculation we refer to [3]

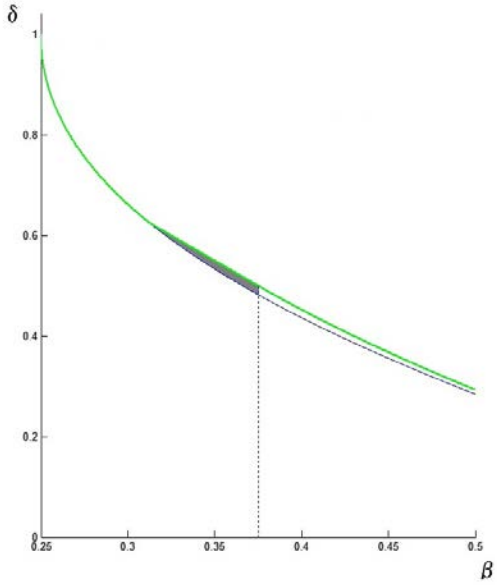


Fig. 2. Improved bound with respect to small prime difference.

References

1. Boneh, D., Durfee, G.:, Cryptanalysis of RSA with Private Key d Less Than $N^{0.292}$, Advances in Cryptology – EUROCRYPT99, Lecture Notes in Computer Science 1592, Berlin: Springer 1999, pp. 1–11
2. Herrmann, M., May, A.,: Attacking Power Generators Using Unravelled Linearization: When Do We Output Too Much?, Proceedings of the 15th International Conference on the Theory and Application of Cryptology and Information Security: Advances in CryptologyIn Advances in Cryptology (Asiacrypt 2009), Lecture Notes in Computer Science 5912, Heidelberg:Springer 2009, pp. 487–504
3. Herrmann, M., May, A.,: Maximizing Small Root Bounds by Linearization and Applications to Small Secret Exponent RSA, In Practice and Theory in Public Key Cryptography (PKC 2010), Lecture Notes in Computer Science 6056, Berlin:Springer-Verlag 2010, pp. 53–69
4. Nick Howgrave-Graham, N.: Finding small roots of univariete modular equations revisited, In Cryptology and Coding, Lecture Notes in Computer Science 1335, Berlin:Springer-Verlag 1997, pp. 131–142
5. Wiener, M.: Cryptanalysis of short RSA secret exponents, IEEE Transactions on Information Theory **36**, 553–558 (1990)
6. De Weger, B.: Cryptanalysis of RSA with small prime difference, Applicable Algebra in Engineering, Communication and Computing, Vol. **13(1)**, 17-28 (2002)

Mars Attacks! Revisited!

Differentially Attack 12 Rounds of the MARS Core and Defeating the Complex MARS Key-Schedule

Michael Gorski, Thomas Knapke, Eik List, Stefan Lucks, and Jakob Wenzel

Bauhaus-University Weimar, Germany
{Michael.Gorski, Thomas.Knapke, Eik.List,
Stefan.Lucks, Jakob.Wenzel}@uni-weimar.de

Introduction The block cipher MARS has been designed by a team from IBM and became one of the five finalists for the AES. A unique feature is the usage of two entirely different round function types. The ”wrapper rounds” are unkeyed, while the key schedule for the ”core rounds” is a slow and complex one, much more demanding then, e.g., the key schedule for the AES. Each core round employs a 62-bit round key. The best attack published so far [KKS00] was applicable to 11 core rounds, and succeeded in recovering some 163 round key bits. But neither did it deal with inverting the key schedule, nor did it provide any other means to recover the remaining 519 round key bits in usage.

Our attack applies to 12 core rounds, needs 2^{252} operations, 2^{65} chosen plaintexts and 2^{69} memory cells. After recovering a limited number of cipher key bits, we deal with the inverse key-schedule to recover the original encryption key. This allows the attacker to easily generate all the round keys in the full.

Recent research provides some amazing advances in the cryptanalysis of the AES, mostly for the 192-bit key and the 256-bit key variants, such as the boomerang attacks on the full-round AES-192 (12 rounds) and AES-256 (14 rounds) [BK09] and the ”practical complexity attack” on AES-256 with up to 13 rounds [BK10]. Since many of these attacks are based on exploiting the AES key-schedule, which is fairly simple, we are interested in having a fresh look at other block ciphers, namely at those with a more complex key-schedule. This makes the AES finalist MARS [BCD$^+$99] a highly relevant subject to study.

MARS consists of a “cryptographic core” in the middle and a “wrapper” surrounding the core. The ”wrapper” is unkeyed, except for two key additions just before the first and after the last wrapper round. Because of its unusual structure MARS differs from the other finalists.

Our Contribution In [KKS00], Kelsey, Kohno and Schneier published an 11-round attack against the MARS core, five forward and six backwards. In contrast to this attack, our attack covers 12 rounds of the MARS core, eight forward and four backwards. The difference results from adding one forward round (Round 3 of our distinguisher) to our attack. Furthermore, we place the two last rounds of the attack in [KKS00] at the beginning of our distinguisher. So we can get more information about the subkeys generated in the first iteration of the key expansion, which are closely linked with the encryption key. Moreover, it allows us to start a meet-in-the-middle attack on the MARS key scheduler. In [KKS00] they can recover a total of 163 subkey bits, but they do not consider a way to attack the key scheduler with this information. So they cannot recover bits from the encryption key.

Differential Cryptanalysis It is one of the most powerful cryptanalysis techniques applied to symmetric-key block ciphers. and was first presented by Biham and Shamir [BS90] at CRYPTO ’90 to attack DES.

Differential cryptanalysis is a chosen plaintext attack that exploits the high probability of certain occurrences of plaintext differences and differences into the last or first round of the cipher. A particular output difference ΔY occurs with a probability p given a particular input difference ΔX, which goes through some non-linear parts of the cipher (see the full paper for more description details)

Description of MARS MARS is a block cipher with a block length of 128 bits and a variable key length from 128 to 448 bits, in increments of 32 bits. The cipher uses 8×32-bit S-boxes in the Mixing Rounds and 9×32-bit S-boxes in the Core Rounds.
The structure consists of sixteen rounds of keyed transformation that build the "cryptographic core". The core is wrapped with unkeyed mixing rounds and additional key whitening. In total, the MARS structure consists of six different layers, that all operate on 32-bit words. Thus, at the beginning, a 128 bit plaintext block is split into the four words A, B, C, D that are then transformed during the encryption as follows:

1. **Pre-Whitening Layer**: To each of the four words A, B, C, D, a different subkey of 32 bits length is added modulo 2^{32}.
2. **Forward Mixing Layer**: Eight rounds of unkeyed mixing, using the S-box, addition and XOR operations.
3. **Forward Core Layer**: Eight rounds of keyed Feistel cipher. The core layer combines a variety of operations, including S-box lookups, multiplications, additions, XORs, fixed-value rotations and data-dependent rotation.
4. **Backward Core Layer**: Eight rounds of keyed Feistel cipher. The core layer combines a variety of operations, including S-box lookups, multiplications, additions, XORs, fixed-value rotations and data-dependent rotation.
5. **Backward Mixing Layer**: Eight rounds of unkeyed mixing, using the S-box, subtraction and xor operations.
6. **Post-Whitening Layer**: From each of the four words A, B, C, D, a different subkey of 32 bits length is subtracted modulo 2^{32}.

We will focus on the core rounds in the following.

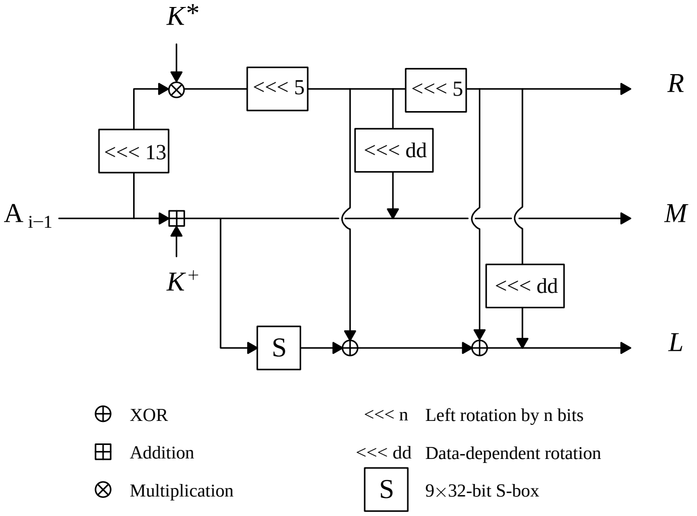


Fig. 1. The MARS core E-function.

The "cryptographic core" of the MARS cipher consists of eight forward and eight backward core rounds. In each round the cipher uses a keyed E-function (E for expansion) which is a combination of multiplication, data-dependent rotations and an S-box lookup. The structure of the Feistel network is depicted in Section 4.1 to visualise our distinguisher. The E-function is shown in Figure 1 (see the full paper for more details).

Differential Attack on 12 Core Rounds of MARS In this section we present the first 12-round differential attack on the reduced MARS core. The first part describes the distinguisher, which is partitioned into four parts. The second part of this section describes the attack to recover the subkey candidates. In the third part we describe the necessary steps to get the secret key. In the fourth part we show the analysis of the attack.

The Distinguisher The distinguisher itself is partitioned in four parts. The first part consists of the Round 1 to 3, where we have to guess both keys of the first Round, both keys of the second Round and the multiplication key of Round 3. Our goal is to reach a difference Δ_1 after two round with the following structure,

$$\Delta_1 = (0, a, b, 0)$$

where a and b are arbitrary differences and 0 is a null difference, which means all 32 bits are set to zero. The second part lasts from Round 4 to 6. Here we use a three-round differential where we choose the output difference as follows:

$$\Delta_2 = (0, 0, 0, (?^7, 0^{15}, ?^{10}))$$

where a 0 stands for a null bit respectively for a null word and a ? stands for an unknown bit difference. The reason for our possibility to choose any difference can be found in the description in the full paper. The third part of our distinguisher lasts from Round 7 to 9. Here we use a three-round differential with a probability nearly one third induced by the binomial distribution and the non-null part of the difference will never seen as the input of the E-function within the Rounds 7 to 9. This leads in an output difference Δ_3 after Round 9.

$$\Delta_3 = ((?^6, a, 0^6, ?^{19}), 0, 0, 0)$$

The last part of the distinguisher lasts from Round 10 to 12. In Round 12 we want to recover the a bit and the six following zero bits of the fourth word of Δ_5. Additionally we create a table for the last round to reduce the amount of subkey candidates. The Distinguisher leads us to a total of 186 subkey bits. For detailed information about the specific differences see the full paper.

The Attack In this section we describe the way to get from the chosen plaintexts to the right subkey candidates. The attack covers 12 rounds of the MARS core, eight forward and four backwards. For this attack we use 2^{56} batches with 302 texts each. This results in a total of 2^{65} chosen plaintexts. For each of the 2^{154} subkey candidates of the first three rounds we have to do the following steps:

1. Choose 2^{56} arbitrary differences as the output of Round 3 (which is the difference described as $(0, a, b, 0)$ from Section 1).
2. Partially decrypt the difference $(0, a, b, 0)$ from the output of Round 3 to determine the input difference (A, B, C, D) of the distinguisher
3. Create 2^{56} batches with 302 texts each where the difference between each of two batches is (A, B, C, D).
4. Encrypt all plaintexts and store the resulting 2^{65} ciphertexts.
5. Partially decrypt all ciphertexts with each of the 2^{32} subkey candidates for Round 12 and extract bit a for each ciphertext.
6. Build 2^{56} 302-bit strings of the bits a for each batch.
7. Sort the resulting bit strings in order of the chosen plaintexts.
8. Compare the bit strings pairwise to identify the correct subkey candidates.

The Key Recovery Step / Key Expansion The key expansion of MARS expands the secret key of n 32-bit words to a total of 40 subkeys of 32-bit words. The key expansion uses a temporary array T to hold the internal state of the transformed subkeys. The array T is initialized by

$$T[0 \dots n-1] = k[0 \dots n-1], \quad T[n] = n, \quad T[n+1 \dots 14] = 0$$

where $n = 8$ (for 256-bit keys) and k is an array with the secret key. The key expansion repeats the following steps four times, where each iteration produces 10 subkeys:

1. **Linear transformation:** The array T is transformed by

$$\text{for } i = 0, \dots, 14, \quad T[i] = T[i] \oplus ((T[i-7 \bmod 15] \oplus T[i-2 \bmod 15]) \lll 3) \oplus (4i + j)$$

2. **Four stirring rounds:** The array T is stirred using four rounds of type-1 Feistel network

$$\text{for } i = 0, \dots, 14, \quad T[i] = (T[i] + S[\text{low 9 bits of } T[i-1 \bmod 15]]) \lll 9$$

3. **Storing keys:** The next 10 keys are stored in the array of subkeys K

$$\text{for } i = 0, \dots, 9, \quad K[10j + i] = T[4i \bmod 15]$$

where $j \in \{0, \dots, 3\}$.
4. **Modification of multiplication keys:** To avoid weak keys, the subkeys that are used for multiplications are modified in an additional step where sequences of 10 or more equal bits are modified. The details of this operation can be found the specification of MARS [BCD$^+$99].

In this section, we are going to mount a meet-in-the-middle attack on the key schedule of MARS. In the forward step we will guess 210 of 256 bits of the secret key, and additionally 8 bits after the linear transformation. After the linear transformation we can perform two stirring rounds and know all bits that go in the S-box in these two rounds. The stirring rounds use additions to transform the words $T[i]$. In each word $T[i]$ that is transformed in these rounds, there is a continuous sequence of some unknown bits that can lead to unpredictable carry bits after the additions of the stirring rounds. For each of the 23 additions we have to consider in the forward step, we have to execute the forward step twice.

In the backward step we use four of the subkeys from the result of our distinguisher, K_4^+, K_5^*, K_6^+ and K_9^* as known input in the backward step. We invert the modification of multiplication with the help of a lookup table that stores the projection results all 2^{32} possible values $K[i]$ and all possible 2^5 rotation values of the patterns. For detailed information about the specific differences see the full paper. In each stirring round we have to guess the low nine bits from the word $T[i-1]$ that are used as input to the S-box and modify one word $T[i]$. In the middle of the MITM attack on the key expansion, we can compare 27 bits of $T[1]$, 27 bits of $T[5]$, 32 bits of $T[6]$ and 21 bits of $T[9]$. So we can compare a total of 107 bits.

Analysis of the Attack For the distinguisher we use 2^{56} batches of 302 texts each and for the difference after Round 4 ($\Delta 2$, see Section 1 for further details) we use nine zeroes. The amount of subkey candidates for our distinguisher is 2^{186}. The probability for a right subkey candidate (i.e., a right bit string) is 2^{-191}, which means we expect one right subkey, and expect only few false positive candidates that we can test by hand. The probability for a false positive subkey candidate is $2^{186} \cdot 2^{-191} = 2^{-5}$. The effort for creating and sorting the table of the last round of the distinguisher is $2^{165} + 165 \cdot 2^{165} \approx 2^{173}$. The effort of the distinguisher is $2^{56} \cdot 302 \cdot 2^{186} \cdot 3.05 = 2^{251.85}$. If we consider the additive effort for the creation of the table for the last Round, the effort is increased to $2^{251.85} + 2^{173} \approx 2^{252}$.

The effort of inverting a multiplication key (last part of the key scheduler) can be reduced (in relation to brute force) with the help of a table with all possible input values T_i and the resulting output values at K_j^*. We found out, that some keys K_j result from the same value T_i. So we tested all possible values of T_i to find the maximum amount of combinations of rotation values and values T_i that can generate the same subkey K_j. We found a subkey that could be generated by 102 such combinations. A table lookup may result in 102 outputs, what increases the resulting effort by 2^7.

The key recovery step is divided in two parts, the forward step and the backward step. This step includes the initialization, the linear transformation and two stirring rounds. The effort for the forward round is $2^{210} \cdot 2^8 \cdot 2^{23} = 2^{241}$. The backward step of this meet-in-the-middle attack includes the inverting of the multiplication key creation and two stirring rounds. The total effort for the backward step is $2^{14} \cdot 2^{27} \cdot 2^{27} + 2^{37} \approx 2^{68}$. In this attack we compare 107 bits in the middle of the four stirring rounds. Thus, the probability of finding two matching bit strings is 2^{-107}. If we combine the efforts of the forward step and the backward step we expect to find $2^{241} \cdot 2^{68} \cdot 2^{-107} = 2^{202}$ possible candidates for the 210 bits of the secret key with an total effort of 2^{202}. After Step 2 we got 210 bit of the encryption key with an effort of 2^{202}. This allow us to obtain the rest of the secret key via brute force, which means the total effort for Step 3 is $2^{202} \cdot 2^{46} = 2^{248}$. This is much faster than obtaining the whole secret key via brute force.

Conclusion In this paper we describe a differential attack on 12 core rounds of the AES candidate cipher MARS based on an attack on 11 rounds of the MARS core that was proposed by [KKS00]. We show that the original attack did not allow the attacker to retrieve information about the encryption key from the gathered subkey material. Our attack improves the original attack, as it covers one additional core round and retrieves subkey information that can be used to recover the encryption key. For this purpose we propose a meet-in-the-middle attack that allows us to invert the key expansion of MARS more efficiently than exhaustive search.

References

[BCD+99] Carolynn Burwick, Don Coppersmith, Edward D'Avignon, Rosario Gennaro, Shai Halevi, Charanjit Jutla, Stephen M. Matyas Jr, Luke O'Connor, Mohammad Peyravian, David Stafford, and Nevenko Zunic. MARS - A Candidate Cipher for AES. *NIST AES Proposal*, 1999.

[BK09] Alex Biryukov and Dmitry Khovratovich. Related-key Cryptanalysis of the Full AES-192 and AES-256. Cryptology ePrint Archive, Report 2009/317, 2009. http://eprint.iacr.org/.

[BK10] Alex Biryukov and Dmitry Khovratovich. Feasible Attack on the 13-round AES-256. Cryptology ePrint Archive, Report 2010/257, 2010. http://eprint.iacr.org/.

[BS90] Eli Biham and Adi Shamir. Differential Cryptanalysis of DES-like Cryptosystems. In Menezes and Vanstone [MV91], pages 2–21.

[BS93] Eli Biham and Adi Shamir. *Differential Cryptanalysis of the Data Encryption Standard*. Springer-Verlag, 1993.

[DR02] Joan Daemen and Vincent Rijmen. *The Design of Rijndael: AES - The Advanced Encryption Standard*. Springer, 2002.

[KKS00] John Kelsey, Tadayoshi Kohno, and Bruce Schneier. Amplified Boomerang Attacks Against Reduced-Round MARS and Serpent. In *Fast Software Encryption*, pages 75–93, 2000.

[KS00] John Kelsey and Bruce Schneier. MARS Attacks! Preliminary Cryptanalysis of Reduced-Round MARS Variants. In *AES Candidate Conference*, pages 169–185, 2000.

[MV91] Alfred Menezes and Scott A. Vanstone, editors. *Advances in Cryptology - CRYPTO '90, 10th Annual International Cryptology Conference, Santa Barbara, California, USA, August 11-15, 1990, Proceedings*, volume 537 of *Lecture Notes in Computer Science*. Springer, 1991.

[NIS00] NIST. A Request for Candidate Algorithm Nominations for the AES. 2000. http://www.nist.gov/aes/.

[Pes09] Alexander Pestunov. Differential Cryptanalysis of the MARS Block Cipher. *Prikladnaya Diskretnaya Matematika*, pages 56–63, 2009. http://mi.mathnet.ru/pdm157.

Group Homomorphic Encryption and Beyond: Characterizations, Impossibility Results, and Applications

Frederik Armknecht[1], Stefan Katzenbeisser[2], and Andreas Peter[2]

[1] Universität Mannheim, Germany
[2] Technische Universität Darmstadt, Germany.

1 Introduction

Homomorphic encryption schemes support computation on encrypted data. Such schemes are of particular interest for various applications, such as Outsourcing of Computation [14], Electronic Voting [2, 6, 8, 9], Private Information Retrieval [26], Oblivious Polynomial Evaluation [29], or Multiparty Computation [7].

The most prominent homomorphic encryption schemes, e.g., ElGamal [13], Paillier [32], Damgård-Jurik [12], are homomorphic with respect to a single algebraic operation. That is, the plaintext space forms a group $(G, \circ)$ and, given encryptions of $m, m' \in G$, one can efficiently and securely compute an encryption of $m \circ m'$ without revealing m and m'. We will call such schemes *group homomorphic* encryption schemes. Although fully homomorphic schemes [5, 15, 16, 37, 39], i.e., schemes that allow one to evaluate any circuit over encrypted data without being able to decrypt, provide a much higher flexibility compared to group homomorphic schemes, the investigation of the latter still represents an important research topic:

1. The majority of existing homomorphic schemes are group homomorphic and there are still many open questions regarding these schemes.
2. For practical applications there is currently no alternative to such schemes.[3]
3. Many constructions of schemes that support more than a single algebraic operation are in particular group homomorphic as well (e.g., [1, 4]).
4. A comprehensive understanding of group homomorphic schemes leads to a better understanding of schemes that are homomorphic in a more general sense, since the underlying structures are very similar. (This is what we focus on in the talk!)

Over the last decades, a variety of different approaches (and according hardness assumptions and proofs of security) has been investigated for constructing group homomorphic schemes, such as the Quadratic Residuosity Problem [19], the Higher Residuosity Problem [2], the Decisional Diffie-Hellman Problem [13, 34], and the Decisional Composite Residuosity Class Problem [32, 12]. All these schemes have been investigated separately, resulting in the fact that some of them are better understood than others. In particular, much effort has been devoted to proving existing homomorphic schemes IND-CCA1 secure (being the highest possible security level for a homomorphic scheme). For example, since the introduction of Damgård's ElGamal [11] in 1991, many works addressed the problem of characterizing its IND-CCA1 security [18,

[3] For example, the most efficient implementation [17] of [16] states that the largest variant (for which a security level similar to RSA-1024 is assumed) has a public key of 2.4 GB size and requires about 30 minutes to complete certain operations.

40]. Similarly, while the IND-CPA security of ElGamal is known for a while [38], the quest for a characterization of its IND-CCA1 security has been in the focus for many years. Only in 2010, the quest concerning these two schemes has finally found an end due to [28]. Finding similar characterizations for remaining homomorphic schemes, e.g., Paillier's scheme, is still an open problem.

2 Contribution and Content of the Talk

In the first part of the talk, we briefly present a unified view both in terms of security and design on group homomorphic encryption schemes among which the most prominent encryption schemes such as ElGamal and Paillier can be found. On the one hand, this helps to access the kind of challenges mentioned above more easily (and in fact, to answer open questions) and on the other hand provides a systematic procedure for designing new schemes based on given problems. More precisely, we construct an abstract scheme that represents all group homomorphic encryption schemes and prove its IND-CCA1 security *equivalent* to the hardness of a new abstract problem, called the *Splitting Oracle-Assisted Subgroup Membership Problem* (SOAP), meaning that every scheme occurs as an instantiation of the abstract scheme being IND-CCA1 secure *if and only if* the according instantiation of SOAP is hard. A *characterization* of IND-CPA security through an abstract problem, called the *Subgroup Membership Problem* (SMP) is an immediate byproduct of our results.

As a direct implementation, we can apply our abstract security characterizations to existing homomorphic schemes by looking at the according instantiations, to deal with the IND-CCA1 (resp. IND-CPA) security of these schemes, or to verify existing IND-CCA1 (resp. IND-CPA) security proofs.

Furthermore, our characterizations allow us to derive impossibility results. For instance, we show that there cannot exist an IND-CPA secure group homomorphic encryption scheme when the ciphertexts form a linear subspace of $\mathbb{F}^n$ for some prime field $\mathbb{F}$, and the output distribution of the encryption algorithm is computationally indistinguishable from the uniform distribution. This partly answers an open question whether using linear codes as ciphertext spaces yield more efficient constructions (see [16]).

Another utilization of our results is a systematic approach for constructing provably secure group homomorphic schemes. By using our abstract scheme and a concrete instantiation of SOAP resp. SMP, one can directly specify a homomorphic scheme that is IND-CCA1 resp. IND-CPA secure if and only if the respective problem is hard.

As a first example, we consider the *k-linear problem* [23, 36] which is an alternative to DDH in groups where DDH is easy, e.g., in bilinear groups [24]. Since its introduction, it is a challenge to construct cryptographic protocols whose security is based on the k-linear problem (e.g., [3, 20, 23, 25, 27, 30, 36]). Following this task, we present the first homomorphic scheme that is based on the k-linear problem for $k > 2$ ($k = 1$ is ElGamal [13], $k = 2$ is Linear Encryption [3]). In addition, we introduce a *new* k-problem (an instantiation of SOAP) that we prove to be hard in the generic group model and to have the same progressive property as the k-linear problem. This result might be of independent interest as it can be used to construct new cryptographic protocols with unique features. For instance, we give the first homomorphic scheme that can be instantiated with groups where DDH is easy (e.g., bilinear groups) and is nevertheless provably secure in terms of IND-CCA1 due to the new k-problem.

The second example is motivated by the main result of [22] stating that one can efficiently construct IND-CCA2 secure encryption schemes from any IND-CPA secure homomorphic encryption scheme whose ciphertext group is *cyclic*. The existence of such schemes was an open question. We positively answer this question by constructing such a scheme and prove it secure under a known problem introduced in [31].

In the second part of the talk, we demonstrate the impact of our results to schemes that are homomorphic in a more general sense (such as fully homomorphic schemes), we first identify a certain structure that all existing homomorphic schemes have in common. In fact, this structure is the key ingredient to our IND-CPA characterization of group homomorphic schemes and allows us to extend our results to more general cases such as fully homomorphic schemes. All currently known fully homomorphic schemes arise by applying a technique that was introduced by Gentry [16] to an underlying (so-called *bootstrappable*) scheme. Interestingly enough, we also show that the IND-CPA security of such fully homomorphic schemes is equivalent to the *1-way* KDM *security* of the underlying scheme (which roughly means that the scheme remains secure even if the adversary gets to see the bits of the secret key encrypted under the corresponding public key).

In the talk, we give a brief overview on Gentry's technique, show how our results extend to this setting, and explain what consequences this has to existing fully homomorphic schemes.

3 Separation from Other Related Work

Aside from the related work that we have already mentioned in the previous sections, there is a substantial number of papers on the construction of IND-CPA (respectively, IND-CCA1, IND-CCA2) secure encryption schemes. In this regard, we would particularly like to mention the work by Cramer and Shoup [10] who give a generic construction of IND-CPA (respectively, IND-CCA1, IND-CCA2) secure encryption schemes through smooth (respectively, 1-universal, 2-universal) hash proof systems. Furthermore, Peikert and Waters [33] introduce the notion of Lossy Trapdoor Functions (LTFs) and give a generic construction of IND-CCA1 secure encryption schemes from such functions, while Hemenway and Ostrovsky [21] give a generic construction of IND-CCA1 secure group homomorphic encryption schemes through homomorphic hash proof systems, which are known to be constructable, e.g., from the Quadratic Residuosity Problem, the Decisional Diffie-Hellman Problem or the Decisional Composite Residuosity Problem. A somewhat different approach to the construction of IND-CCA1 secure group homomorphic encryption was presented by Prabhakaran and Rosulek [35]. Therein, they build group homomorphic encryption schemes that are secure in an even stronger sense than just being IND-CCA1, namely "homomorphic-CCA" secure.

All these works have in common that they build IND-CCA1 secure schemes from non-interactive assumptions, while we show the IND-CCA1 security equivalent to the hardness of SOAP which then naturally has to be an interactive problem, as IND-CCA1 is. Therefore, we stress that we give *characterizations* of the security of group homomorphic schemes. For all the above mentioned schemes this means that the underlying non-interactive assumption either implies SOAP, or is equivalent to it. In the former case, breaking the underlying assumption would not necessarily break the security of the scheme in question as it is actually equivalent to SOAP which might still be a hard problem. We do not give a generic construction of IND-CCA1 secure group homomorphic schemes from *non-interactive* assumptions. Concerning IND-CPA security on the other hand, this is a completely different story, as we propose the first generic scheme that encompasses *all* group homomorphic encryption schemes and hence is a also a generic way to construct IND-CPA secure group homomorphic schemes from non-interactive assumptions. The latter is due to the fact that the corresponding SMP instance is always non-interactive.

References

1. Frederik Armknecht and Ahmad-Reza Sadeghi. A new approach for algebraically homomorphic encryption. Cryptology ePrint Archive, Report 2008/422, 2008. http://eprint.iacr.org/.
2. J. Benaloh. *Verifiable secret-ballot elections*. PhD thesis, Yale University, New Haven, CT, USA, 1987.
3. Dan Boneh, Xavier Boyen, and Hovav Shacham. Short group signatures. In Matthew K. Franklin, editor, *CRYPTO*, volume 3152 of *Lecture Notes in Computer Science*, pages 41–55. Springer, 2004.
4. Dan Boneh, Eu-Jin Goh, and Kobbi Nissim. Evaluating 2-dnf formulas on ciphertexts. In Joe Kilian, editor, *TCC*, volume 3378 of *Lecture Notes in Computer Science*, pages 325–341. Springer, 2005.
5. Kai-Min Chung, Yael Kalai, and Salil Vadhan. Improved delegation of computation using fully homomorphic encryption. In Tal Rabin, editor, *Advances in Cryptology CRYPTO 2010*, volume 6223 of *Lecture Notes in Computer Science*, pages 483–501. Springer Berlin / Heidelberg, 2010.
6. Josh D. Cohen and Michael J. Fischer. A robust and verifiable cryptographically secure election scheme (extended abstract). In *FOCS*, pages 372–382. IEEE, 1985.
7. Ronald Cramer, Ivan Damgård, and Jesper Buus Nielsen. Multiparty computation from threshold homomorphic encryption. In Birgit Pfitzmann, editor, *EUROCRYPT*, volume 2045 of *Lecture Notes in Computer Science*, pages 280–299. Springer, 2001.
8. Ronald Cramer, Matthew K. Franklin, Berry Schoenmakers, and Moti Yung. Multi-autority secret-ballot elections with linear work. In *EUROCRYPT*, pages 72–83, 1996.
9. Ronald Cramer, Rosario Gennaro, and Berry Schoenmakers. A secure and optimally efficient multi-authority election scheme. In *EUROCRYPT*, pages 103–118, 1997.
10. Ronald Cramer and Victor Shoup. Universal hash proofs and a paradigm for adaptive chosen ciphertext secure public-key encryption. In Lars R. Knudsen, editor, *EUROCRYPT*, volume 2332 of *Lecture Notes in Computer Science*, pages 45–64. Springer, 2002.
11. Ivan Damgård. Towards practical public key systems secure against chosen ciphertext attacks. In Joan Feigenbaum, editor, *CRYPTO*, volume 576 of *Lecture Notes in Computer Science*, pages 445–456. Springer, 1991.
12. Ivan Damgård and Mads Jurik. A generalisation, a simplification and some applications of paillier's probabilistic public-key system. In Kwangjo Kim, editor, *Public Key Cryptography*, volume 1992 of *Lecture Notes in Computer Science*, pages 119–136. Springer, 2001.
13. Taher El Gamal. A public key cryptosystem and a signature scheme based on discrete logarithms. In *CRYPTO*, pages 10–18, 1984.
14. Rosario Gennaro, Craig Gentry, and Bryan Parno. Non-interactive verifiable computing: Outsourcing computation to untrusted workers. In Tal Rabin, editor, *CRYPTO*, volume 6223 of *Lecture Notes in Computer Science*, pages 465–482. Springer, 2010.
15. Craig Gentry. *A fully homomorphic encryption scheme.* PhD thesis, Stanford University, 2009. crypto.stanford.edu/craig.
16. Craig Gentry. Fully homomorphic encryption using ideal lattices. In Michael Mitzenmacher, editor, *STOC*, pages 169–178. ACM, 2009.
17. Craig Gentry and Shai Halevi. *Implementing Gentry's Fully-Homomorphic Encryption Scheme*, August 2010. https://researcher.ibm.com/researcher/files/us-shaih/fhe-implementation.pdf.
18. Kristian Gjøsteen. A new security proof for damgård's elgamal. In David Pointcheval, editor, *CT-RSA*, volume 3860 of *Lecture Notes in Computer Science*, pages 150–158. Springer, 2006.
19. Shafi Goldwasser and Silvio Micali. Probabilistic encryption. *J. Comput. Syst. Sci.*, 28(2):270–299, 1984.
20. Jens Groth, Rafail Ostrovsky, and Amit Sahai. Non-interactive zaps and new techniques for nizk. In Cynthia Dwork, editor, *CRYPTO*, volume 4117 of *Lecture Notes in Computer Science*, pages 97–111. Springer, 2006.

21. Brett Hemenway and Rafail Ostrovsky. Lossy trapdoor functions from smooth homomorphic hash proof systems. *Electronic Colloquium on Computational Complexity (ECCC)*, 16:127, 2009.
22. Brett Hemenway and Rafail Ostrovsky. Homomorphic encryption over cyclic groups implies chosen-ciphertext security. Cryptology ePrint Archive, Report 2010/099, 2010. http://eprint.iacr.org/.
23. Dennis Hofheinz and Eike Kiltz. Secure hybrid encryption from weakened key encapsulation. In Alfred Menezes, editor, *CRYPTO*, volume 4622 of *Lecture Notes in Computer Science*, pages 553–571. Springer, 2007.
24. Antoine Joux and Kim Nguyen. Separating decision diffie-hellman from computational diffie-hellman in cryptographic groups. *J. Cryptology*, 16(4):239–247, 2003.
25. Eike Kiltz. Chosen-ciphertext security from tag-based encryption. In Shai Halevi and Tal Rabin, editors, *TCC*, volume 3876 of *Lecture Notes in Computer Science*, pages 581–600. Springer, 2006.
26. Eyal Kushilevitz and Rafail Ostrovsky. Replication is not needed: Single database, computationally-private information retrieval. In *FOCS*, pages 364–373, 1997.
27. Allison B. Lewko and Brent Waters. Efficient pseudorandom functions from the decisional linear assumption and weaker variants. In Ehab Al-Shaer, Somesh Jha, and Angelos D. Keromytis, editors, *ACM Conference on Computer and Communications Security*, pages 112–120. ACM, 2009.
28. Helger Lipmaa. On the cca1-security of elgamal and damgård's elgamal. In *Proceedings of Inscrypt 2010*. Springer, 2010. http://research.cyber.ee/∼lipmaa/papers/lip10/. To appear.
29. Moni Naor and Benny Pinkas. Oblivious polynomial evaluation. *SIAM J. Comput.*, 35(5):1254–1281, 2006.
30. Moni Naor and Gil Segev. Public-key cryptosystems resilient to key leakage. In Shai Halevi, editor, *CRYPTO*, volume 5677 of *Lecture Notes in Computer Science*, pages 18–35. Springer, 2009.
31. Juan Manuel González Nieto, Colin Boyd, and Ed Dawson. A public key cryptosystem based on a subgroup membership problem. *Des. Codes Cryptography*, 36(3):301–316, 2005.
32. Pascal Paillier. Public-key cryptosystems based on composite degree residuosity classes. In *EUROCRYPT*, pages 223–238, 1999.
33. Chris Peikert and Brent Waters. Lossy trapdoor functions and their applications. In Cynthia Dwork, editor, *STOC*, pages 187–196. ACM, 2008.
34. Manoj Prabhakaran and Mike Rosulek. Homomorphic encryption with cca security. In Luca Aceto, Ivan Damgård, Leslie Ann Goldberg, Magnús M. Halldórsson, Anna Ingólfsdóttir, and Igor Walukiewicz, editors, *ICALP (2)*, volume 5126 of *Lecture Notes in Computer Science*, pages 667–678. Springer, 2008.
35. Manoj Prabhakaran and Mike Rosulek. Homomorphic encryption with cca security. In Luca Aceto, Ivan Damgård, Leslie Ann Goldberg, Magnús M. Halldórsson, Anna Ingólfsdóttir, and Igor Walukiewicz, editors, *ICALP (2)*, volume 5126 of *Lecture Notes in Computer Science*, pages 667–678. Springer, 2008.
36. Hovav Shacham. A cramer-shoup encryption scheme from the linear assumption and from progressively weaker linear variants. Cryptology ePrint Archive, Report 2007/074, 2007. http://eprint.iacr.org/.
37. Nigel P. Smart and Frederik Vercauteren. Fully homomorphic encryption with relatively small key and ciphertext sizes. In Phong Q. Nguyen and David Pointcheval, editors, *Public Key Cryptography*, volume 6056 of *Lecture Notes in Computer Science*, pages 420–443. Springer, 2010.
38. Yiannis Tsiounis and Moti Yung. On the security of elgamal based encryption. In Hideki Imai and Yuliang Zheng, editors, *Public Key Cryptography*, volume 1431 of *Lecture Notes in Computer Science*, pages 117–134. Springer, 1998.
39. Marten van Dijk, Craig Gentry, Shai Halevi, and Vinod Vaikuntanathan. Fully homomorphic encryption over the integers. In Henri Gilbert, editor, *EUROCRYPT*, volume 6110 of *Lecture Notes in Computer Science*, pages 24–43. Springer, 2010.
40. J. Wu and D.R. Stinson. On the security of the elgamal encryption scheme and damgards variant. Cryptology ePrint Archive, Report 2008/200, 2008. http://eprint.iacr.org/.

ON THE ROLE OF EXPANDER GRAPHS IN KEY PREDISTRIBUTION SCHEMES FOR WIRELESS SENSOR NETWORKS - EXTENDED ABSTRACT

MICHELLE KENDALL, KEITH MARTIN

Abstract. The topic of key predistribution schemes for wireless sensor networks has been widely studied from a variety of perspectives. In particular, key predistribution schemes have been proposed based on expander graph theory, and it has been claimed that good expansion properties are necessary for optimal networks. We examine the role of expander graphs in wireless sensor networks, demonstrate flaws in previous suggestions about product graph expansion and optimality, show that the success of many schemes is related to their good expansion properties, and explain the extent to which expander graphs can be said to provide optimal solutions.

In the context of key predistribution schemes (KPS) for wireless sensor networks (WSN), the topic of expander graphs was introduced in 2006 from two different angles. Camtepe et al [2] showed that an expander graph construction could be used as a template for KPS, and Ghosh [5] made claims linking the necessity of good expansion to 'optimal' WSN solutions. We show that Ghosh's claims are flawed but explain the reason why expansion properties are beneficial for WSNs. We then study precisely where these expansion properties are needed and where they can be controlled in order to get close to an optimally secured and functioning WSN.

We begin by introducing the relevant terminology and concepts in section 1. In section 2 we outline Ghosh's claim and show by means of a counter-example that his conclusion is misdirected towards expansion in product graphs rather than intersection graphs. We consider in section 3 how to maximise the probability of a high expansion parameter in the intersection graph, and conclude in section 4 with comments on the relationship between expansion and optimality in WSNs.

1. Background

1.1. KPS.

We consider the deployment of large numbers of small sensor devices or 'nodes' with the aim of creating a network for the communication of data. This should be secured by cryptographic keys stored on the nodes before deployment, or *predistributed*. Since the nodes are resource-constrained, the aim is to minimise key storage whilst maximising the connectivity and resilience of the network.

Resilience is a measure of the network's ability to withstand damage as an adversary compromises nodes, learning the keys which they store. We measure it by the parameter fail_s, which is the probability that a random link between two uncompromised nodes is rendered insecure because the adversary knows the appropriate keys after compromising a set of s nodes elsewhere in the network.

To illustrate the trade-offs required, we consider some trivial examples of KPS. If each node were preloaded with a single key k, then this would require minimal memory and ensure that any pair of nodes could communicate securely. However, there would be minimal resilience against an adversary, as $\mathsf{fail}_s = 1$ for all $1 \leq s < n$, where n is the total number of nodes in the network.

Another trivial approach is to pick a unique key for each pair of nodes. This has maximal resilience against an adversary as $\mathsf{fail}_s = 0$ for all $0 \leq s < n$. However, each node would have to store $n - 1$ keys, which is infeasible in large networks.

To find efficient trade-offs between these conflicting parameters, a variety of KPS have been proposed, a survey of which can be found in [1].

1.2. **Graph theory.** We now introduce the relevant graph-theoretic definitions. A *graph* $G = (V, E)$ is a set of vertices V and a set of edges E. We draw a graph of a WSN by representing the nodes as vertices and communciation channels as edges. We say that a graph is *connected* if there is a path (a sequence of edges) between every pair of nodes, and *complete* if there is an edge between every pair of nodes. The *degree* of a node is its number of edges.

To be precise in our analysis, we consider the separate component graphs of a network: the *communication graph* $G = (V, E_G)$ where $(u, v) \in E_G$ if u and v are in communication range, and the *key graph* $H = (V, E_H)$ where $(u, v) \in E_H$ if u and v share a common key.

Notice that in the trivial KPS examples given in section 1.1 both key graphs are complete but the resilience is very different. The value of fail_s is not directly related to the connectivity of the key graph, but instead to the number of keys known to each node as a proportion of the total number of keys used in the network.

Two nodes u and v can *communicate securely* if $(u, v) \in E_G \cap E_H$, that is if they share an edge in the *intersection* graph $G \cap H = (V, E_{G \cap H})$. If nodes do not share an edge in the intersection graph then there are usually ways for them to route messages through intermediary nodes and/or establish new keys, but this requires extra communicational overheads and so it is desirable to minimise the *diameter*, the longest path length between nodes.

Finally, we introduce another way of combining two graphs. The (Cartesian) *product* graph is defined as $G.H = (V \times V, E_{G.H})$, where edges obey the rule: $(uv, u'v') \in E_{G.H}$ if $(u = u'$ or $(u, u') \in E_G)$ AND $(v = v'$ or $(v, v') \in E_H)$.

We will now define expander graphs and consider why their properties are desirable in WSNs.

1.3. **Expander graphs.** The *expansion* of a graph is a measure of the strength of its connectivity. For a thorough survey of expander graphs and their applications, see [6]. The edge-expansion parameter ϵ for a graph $G = (V, E)$ is defined by

$$\epsilon = \min_{S \subset V : |S| \leq \frac{|V|}{2}} \left(\frac{|E(S, \overline{S})|}{|S|} \right)$$

where $|E(S, \overline{S})|$ denotes the number of edges from the set S to $\overline{S}$, vertices not in S.

If $\epsilon = 0$, this implies that there exists a subset $S \subset V$ without any edges connecting it to the rest of the graph, and we conclude that the graph is not connected. Conversely, if the graph is connected then $\epsilon > 0$, hence all connected graphs are ϵ-expander graphs for some positive value of ϵ.

If ϵ is 'small', for example $\epsilon = \frac{1}{100}$, then there exists a set of vertices S which is only connected to the rest of the graph by one edge per 100 nodes in S. This is undesirable for a WSN as it makes the set S vulnerable to being 'cut off' from the rest of the network by a small number of attacks or faults and increases the communication burden on a small set of nodes, quickly draining their batteries. If ϵ is larger, particularly if $\epsilon > 1$, then there is no easy way to disconnect large sets of nodes and communicational burdens are more evenly spread. It also ensures that the graph has low diameter, logarithmic in the size of the network, and multiple short paths between nodes, which is beneficial for schemes like the multipath reinforcement of Chan et al. [3].

Observe that in a finite connected graph $G = (V, E)$, the value of ϵ is bounded:

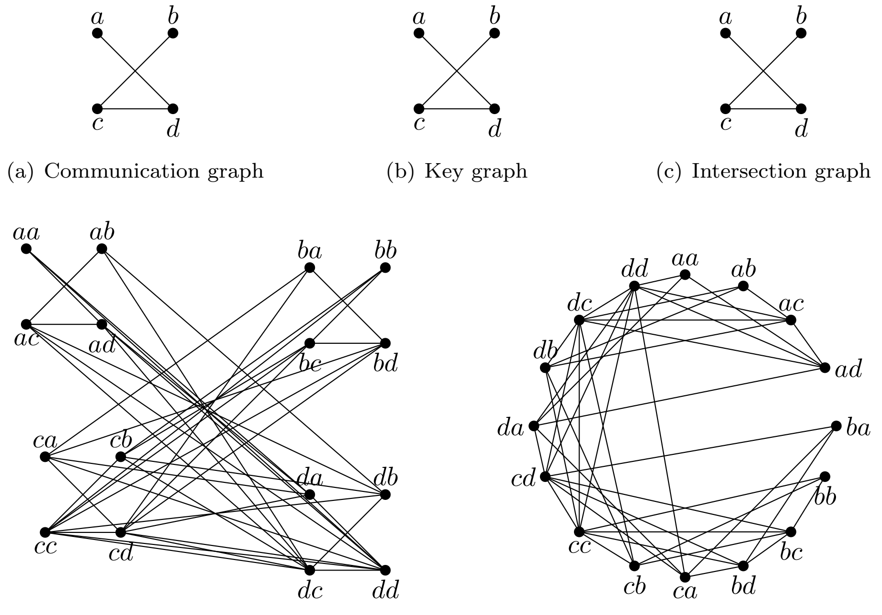
$$0 \leq \epsilon \leq \min_{v \in V} d_G(v) \ , \tag{1}$$

where the upper bound is the smallest degree of any vertex in the graph.

2. Expansion in product graphs

We will now examine Ghosh's claim about expander graphs and WSNs. In his paper 'On optimality of key predistribution schemes for distributed sensor networks', Ghosh claims that in order to optimise the conflicting parameters of large network size, low key storage per node, high connectivity and high resilience, the product graph must have 'good expansion properties' [5]. However, we show by an example that expansion in the product graph is not a helpful measure and that the product graph is unable to capture the required detail to analyse a WSN.

Figure 1 shows a communication graph and a key graph, and their corresponding intersection and product graphs. The product graph is represented in Figure 1(d) in a way which demonstrates its construction, and redrawn in Figure 1(e) for clarity. The communication and key graphs are identical, giving the best possible case for intersection, and the product graph has expansion parameter $\epsilon = \frac{5}{4}$.



(a) Communication graph

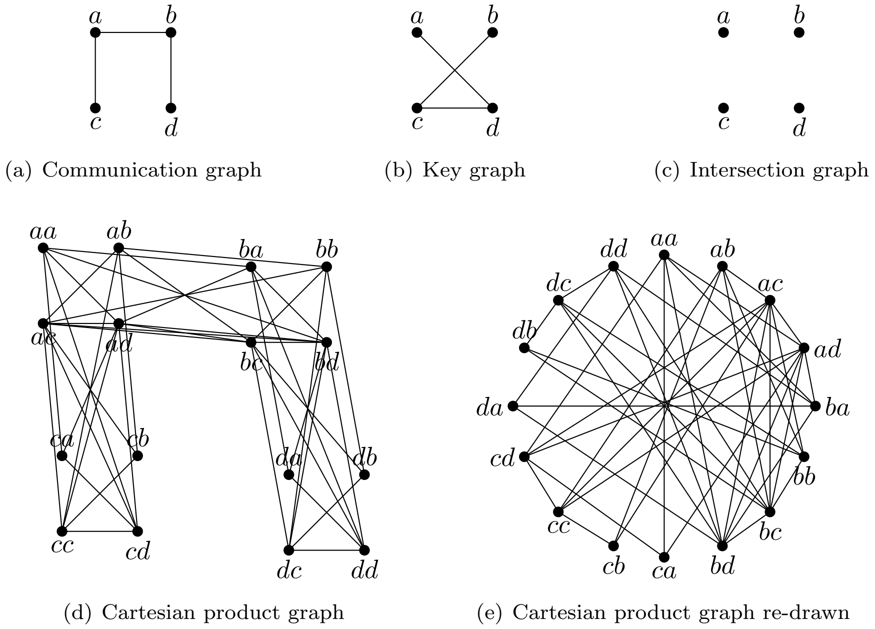
(b) Key graph

(c) Intersection graph

(d) Cartesian product graph

(e) Cartesian product graph re-drawn

Figure 1. The intersection of the component graphs retains all 3 edges



(a) Communication graph

(b) Key graph

(c) Intersection graph

(d) Cartesian product graph

(e) Cartesian product graph re-drawn

FIGURE 2. The intersection of the component graphs has no edges

On the other hand, a different communication graph in Figure 2 results in the intersection graph having no edges (meaning that no secure communication is possible) but the product graph again has expansion parameter $\epsilon = \frac{5}{4}$. Indeed, we can see that the product graphs of Figures 1 and 2 are isomorphic, using a simple bijection to relabel vertices as follows:

Fig. 1(e)		Fig 2(e)
$(a*)$	$\rightarrow$	$(c*)$
$(b*)$	$\rightarrow$	$(d*)$
$(c*)$	$\rightarrow$	$(b*)$
$(d*)$	$\rightarrow$	$(a*)$

This means that all properties of connectivity, expansion, degree, maximum path length etc. are identical between the two product graphs. We see that it is the intersection graph where good expansion is desirable, and that the product graph is not able to capture the details of the intersection properties of its component graphs.

3. EXPANSION IN INTERSECTION GRAPHS

We now consider the expansion properties of intersection graphs, beginning by analysing the degrees of individual nodes. It is easy to see that the degree of a node in the intersection graph $G \cap H$ cannot be larger than its degree in either of the underlying graphs. Indeed, for all $v \in V$, $d_{G\cap H}(v) \leq \min\{d_G(v), d_H(v)\}$. Together with the inequality (1), this shows that

$$\epsilon_{G\cap H} \leq \min_{v\in V}\{d_{G\cap H}(v)\} \quad \text{and} \quad \epsilon_{G\cap H} \leq \min\{\epsilon_G, \epsilon_H\} \ .$$

Therefore it is necessary that G and H have high expansion parameters for $G \cap H$ to be a good expander. If the communication graph is complete then the expansion of the key graph will be preserved in the intersection. However, we usually assume that we have little or no control over the communication graph and model it as random, in which case all that can be done is to make sure that the key graph has as high expansion as possible for given levels of key storage and resilience.

Camtepe et al. [2] and Shafiei et al. [7] propose KPS based on expander graph constructions and demonstrate that these schemes compare well to other well-regarded KPS approaches. Indeed, the success of other KPS approaches is due at least in part to the fact that they also produce key graphs with good expansion parameters for chosen levels of key storage and resilience. For example, a fundamental reason for the success of Eschenauer and Gligor's random KPS [4] is that random graphs are good expanders with high probability [6].

4. Concluding remarks

We conclude with some comments on optimality, since the aim of Ghosh's paper [5] is to optimise the conflicting parameters in WSNs. The word 'optimal' should be used with caution when describing a trade-off of many parameters, since priorities between these parameters will vary in different scenarios. However, we have shown that if we fix levels of low key storage, large network size and high resilience, then the larger the value of ϵ in the intersection graph, the better connected it will be, with lower diameter and fewer weak points.

This shows that in a setting where there is control over the communication graph, the expansion of the intersection graph should be an important consideration in the design of the key graph. If there is no control over the communication graph, a choice of key graph with maximal expansion is likely to be the best possible for given levels of key storage and resilience, as it will maximise the probability of achieving high expansion in the intersection graph.

References

[1] Seyit Ahmet Camtepe and Bulent Yener. Key distribution mechanisms for wireless sensor networks: a survey. *Rensselaer Polytechnic Institute, Computer Science Department, Tech. Rep. TR-05-07*, 2005.

[2] Seyit Ahmet Camtepe, Bulent Yener, and Moti Yung. Expander graph based key distribution mechanisms in wireless sensor networks. In *ICC 06, IEEE International Conference on Communications*, pages 2262–2267, 2006.

[3] Haowen Chan, Adrian Perrig, and Dawn Song. Random key predistribution schemes for sensor networks. In *SP '03: Proceedings of the 2003 IEEE Symposium on Security and Privacy*, page 197, Washington, DC, USA, 2003. IEEE Computer Society.

[4] Laurent Eschenauer and Virgil D. Gligor. A key-management scheme for distributed sensor networks. In *CCS '02: Proceedings of the 9th ACM Conference on Computer and Communications Security*, pages 41–47, New York, NY, USA, 2002. ACM.

[5] Subhas Ghosh. On optimality of key pre-distribution schemes for distributed sensor networks. In Levente Buttyan, Virgil Gligor, and Dirk Westhoff, editors, *Security and Privacy in Ad-Hoc and Sensor Networks*, volume 4357 of *Lecture Notes in Computer Science*, pages 121–135. Springer Berlin / Heidelberg, 2006. 10.1007/11964254$_1$2.

[6] Shlomo Hoory, Nathan Linial, and Avi Wigderson. Expander graphs and their applications. *Bulletin - American Mathematical Society*, 43(4):439–562, 2006.

[7] H. Shafiei, A. Mehdizadeh, A. Khonsari, and M. Ould-Khaoua. A combinatorial approach for key-distribution in wireless sensor networks. In *Global Telecommunications Conference, 2008. IEEE GLOBECOM 2008. IEEE*, pages 1 –5, 30 2008-dec. 4 2008.

An Information-Theoretic and Computational Complexity Security Analysis of a Randomized Stream Cipher Model

Miodrag J. Mihaljević[1] and Hideki Imai[2]

[1]Serbian Academy of Sciences and Arts, Belgrade, Serbia, & Research Center for Information Security (RCIS), Institute of Advanced Industrial Science and Technology (AIST), Japan.
Email: miodragm@turing.mi.sanu.ac.rs.

[2]Faculty of Sciences and Engineering, Chuo University, Tokyo, & Research Center for Information Security (RCIS), Institute of Advanced Industrial Science and Technology (AIST), Japan.

Abstract. This paper yields an information-theoretic and a computational complexity analysis of the security of a generic model of randomized stream ciphers. The analytical expression of the secret key equivocation given the output of the cipher under the chosen plaintext attacking scenario is derived and analyzed. Regarding the computational complexity security evaluation, it is pointed out that the secret key recovery is as hard as decoding of a random linear block code and that the indistinguishability is as hard as certain LPN problem.

Keywords: symmetric cryptography, encryption, homophonic coding, error-correction coding, randomized encryption, stream ciphers, equivocation, LPN problem.

1 Introduction

Randomized symmetric key encryption as an alternative encryption paradigm is considered in [13]. According to [13], the randomized encryption is a procedure which enciphers a message by randomly choosing a ciphertext from a set of ciphertexts corresponding to the message under the current encryption key, and the following is claimed, [13]: ”At the cost of increasing the required bandwidth, randomized encryption procedures may achieve greater cryptographic security than their deterministic counterparts ...”. In [3], a pseudorandom number generator based on the Learning from Parity with Noise (LPN) problem (related to the hardness of decoding a random linear code) has been reported. Informally, the LPN problem can be considered as the problem of solving a system of linear equations corrupted by noise. or a problem of decoding a linear block code. Recently a number of randomized symmetric key encryption techniques has been reported [6], [10], [11] [1] and [12]. In [6], a probabilistic private-key encryption scheme named LPN-C whose security can be reduced to the hardness of the LPN problem has been proposed and considered. In [1] a symmetric encryption scheme similar to the one reported in [6] is reported and its security and implementation complexity are analyzed. The symmetric encryption schemes reported in [6] and [1] appears as interesting and stimulating for further considerations (having in mind improvements as well) particularly because the security is related to the recognized hard (LPN) problem. Following the encryption approaches recently reported in [10] - [12], this paper considers and analyzes from security point of view a generic model of randomized stream ciphers.

Summary of the Results.

This paper yields an analysis of security of a model of randomized stream cipher based on joint employment of pseudorandomness, randomness and dedicated coding. The considered scheme sequentially encrypts ℓ-bit plaintext vectors into n-bit, $n > \ell$, ciphertext vectors employing: (i) a keystream generator seeded by k-bit secret key, (ii) $m - \ell$, $\ell < m < n$, balanced random bits where ones and zeros appear with the same probability equal to 1/2, (iii) n biased random bits where ones appear with the probability $p < 1/2$, and (iv) two linear encoding schemes for dedicated homophonic-like and error correction encoding. The security analysis has been performed assuming the chosen plaintext attack. The information-theoretic security evaluation was focussed towards the posterior uncertainty on the secret key. The equivocation of the secret key has been derived and analyzed. The equivocation expression shows that it can be kept to a nonzero value assuming appropriate selection of the encryption parameters $m-\ell$, n and p, when the sample available for cryptanalysis is limited. The previous imply that the scheme has potential of providing residual uncertainty on the secret key under certain conditions. Particularly, it is shown that the equivocation is a monotony increasing function of the parameter p and that it achieves its maximal value equal to k when $p = 1/2$ (which is not a surprising outcome but it indicates the correctness of the entire analysis, as well). On the other hand it is shown that the equivocation is a monotony decreasing function of the parameter n and the sample dimension τ, and that limes of equivocation when τ(or n) $\rightarrow \infty$ tends to 0, implying that, when enough long sample is available for cryptanalysis, the uncertainty reduces to zero, i.e. the secret key can be correctly recovered. On the other hand, the previous implies only recoverability of the secret key but does not tell us how complex the recovery is. Accordingly the considered encryption scheme is analyzed from computational complexity security point of view, as well. The performed evaluation of the secret key recovery implies that it is as hard as decoding of a random linear block code after a binary symmetric channel with the additive noise (cross-over probability) parameter ϵ equal to $\frac{1-(1-2p)^{(m-\ell)/2}}{2}$, and also that the indistinguishability is as hard as the LPN problem with the noise equal to ϵ.

The analysis performed imply that the considered encryption paradigm provides a framework for design of provably secure stream ciphers which can provide low implementation complexity as well (noting that the implementation issues are out of the scope of this paper). Accordingly, the given analysis provides particular guidelines for design of randomized stream ciphers which fulfil certain requirements regarding the security and implementation/communications overhead.

2 A Framework of Randomized Stream Cipher

We consider the randomized stream ciphers framework displayed in the following figure.

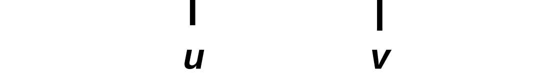


Fig. 1. A generic randomized stream cipher encryption.

For algebraic description of the considered encryption, when we consider encryption of a sequence of vectors at the time instances $t = 1, 2, ..., \tau$, the following notation is employed:
- $\mathbf{a}_t^{(\ell)}$ is a known ℓ-dimensional binary vector at the time instance t;
- $f_t^{(t)}(\mathbf{k})$ is the keystream generator output segment of length n generated at the time instance t;
- $\mathbf{u}_t^{(m-\ell)}$ is a realization of $(m-\ell)$-dimensional binary random variable $\mathbf{U}_t^{(m-\ell)}$, at the time instance t, such that $\Pr(\mathbf{U}_t^{(m-\ell)} = \mathbf{u}_t^{(m-\ell)}) = \frac{1}{2^{m-\ell}}$;
- $\mathbf{v}_t^{(n)}$ is a realization of n-dimensional binary random variable $\mathbf{V}_t^{(n)}$ at the time instance t such that

$\Pr(\mathbf{V}_t^{(n)} = \mathbf{v}_t^{(n)}) = p^{w_t}(1-p)^{n-w_t}$, $p < 1/2$, and $w_t = Hwt(\mathbf{v}_t^{(n)})$ denotes the Hamming weight of the vector $\mathbf{v}_t^{(n)}$.

Accordingly, the ciphertext vectors $\mathbf{z}_t^{(n)}$, $t = 1, 2, ..., \tau$, are specified by the following:

$$\mathbf{z}_t^{(n)} = [\mathbf{a}_t^{(\ell)} || \mathbf{u}_t^{(m-\ell)}]\mathbf{G} \oplus f_t^{(n)}(\mathbf{k}) \oplus \mathbf{v}_t^{(n)} \ , \ t = 1, 2, ..., \tau \ . \tag{1}$$

The corresponding decryption process is as follows:

$$\mathbf{a}_t^{(\ell)} = tcat\{[ECC^{-1}(\mathbf{z}_t^{(n)} \oplus f_t^{(n)}(\mathbf{k}))]\mathbf{G}_H^{-1}\} \ , \ t = 1, 2, ..., \tau \ , \tag{2}$$

where $tcat\{\cdot\}$ is the operator of truncation to the first ℓ bits, $ECC^{-1}(\cdot)$ denotes the decoding operator of the employed error correction code (ECC) with the generator matrix $\mathbf{G}_{ECC}$, and $\mathbf{G}_H^{-1}$ is the inverse matrix of $\mathbf{G}_H$ assuming that $\mathbf{G}_H \cdot \mathbf{G}_{ECC} = \mathbf{G}$. Particulary note that the matrix $\mathbf{G}_H$ corresponds to a homophonic encoding approach (see [7] and [9], for example).

3 An Information-Theoretic Analysis of the Security

Preliminaries. When $\tau = 1$ and omitting (for simplicity of the notations) the index t the posterior probability of the secret key given a sample $\mathbf{z}^{(n)}$ can be expressed as follows.

$$\Pr(\mathbf{K} = \mathbf{k} | \mathbf{Z}^{(n)} = \mathbf{z}^{(n)}) = \frac{\Pr(\mathbf{Z}^{(n)} = \mathbf{z}^{(n)} | \mathbf{K} = \mathbf{k}) \Pr(\mathbf{K} = \mathbf{k})}{\sum_{\mathbf{k}} \Pr(\mathbf{Z}^{(n)} = \mathbf{z}^{(n)} | \mathbf{K} = \mathbf{k}) \Pr(\mathbf{K} = \mathbf{k})} \ ,$$

and when all the keys are equiprobable

$$\Pr(\mathbf{K} = \mathbf{k} | \mathbf{Z}^{(n)} = \mathbf{z}^{(n)}) = \frac{\Pr(\mathbf{Z}^{(n)} = \mathbf{z}^{(n)} | \mathbf{K} = \mathbf{k})}{\sum_{\mathbf{k}} \Pr(\mathbf{Z}^{(n)} = \mathbf{z}^{(n)} | \mathbf{K} = \mathbf{k})} \ ,$$

On the other hand we have the following.

$$\Pr(\mathbf{Z}^{(n)} = \mathbf{z}^{(n)} | \mathbf{K} = \mathbf{k}) = \frac{\Pr(\mathbf{Z}^{(n)} = \mathbf{z}^{(n)}, \mathbf{K} = \mathbf{k})}{\Pr(\mathbf{K} = \mathbf{k})} = \frac{\sum_{\mathbf{u}^{(m-\ell)}} \Pr(\mathbf{Z}^{(n)} = \mathbf{z}^{(n)}, \mathbf{K} = \mathbf{k}, \mathbf{U}^{(m-\ell)} = \mathbf{u}^{(m-\ell)})}{\Pr(\mathbf{K} = \mathbf{k})}$$

$$= \frac{\sum_{\mathbf{u}^{(m-\ell)}} \Pr(\mathbf{Z}^{(n)} = \mathbf{z}^{(n)} | \mathbf{K} = \mathbf{k}, \mathbf{U}^{(m-\ell)} = \mathbf{u}^{(m-\ell)}) \Pr(\mathbf{K} = \mathbf{k}, \mathbf{U}^{(m-\ell)} = \mathbf{u}^{(m-\ell)})}{\Pr(\mathbf{K} = \mathbf{k})}$$

$$= \frac{\sum_{\mathbf{u}^{(m-\ell)}} \Pr(\mathbf{Z}^{(n)} = \mathbf{z}^{(n)} | \mathbf{K} = \mathbf{k}, \mathbf{U}^{(m-\ell)} = \mathbf{u}^{(m-\ell)}) \Pr(\mathbf{U}^{(m-\ell)} = \mathbf{u}^{(m-\ell)} | \mathbf{K} = \mathbf{k}) \Pr(\mathbf{K} = \mathbf{k})}{\Pr(\mathbf{K} = \mathbf{k})}$$

$$= \sum_{\mathbf{u}^{(m-\ell)}} \Pr(\mathbf{Z}^{(n)} = \mathbf{z}^{(n)} | \mathbf{K} = \mathbf{k}, \mathbf{U}^{(m-\ell)} = \mathbf{u}^{(m-\ell)}) \Pr(\mathbf{U}^{(m-\ell)} = \mathbf{u}^{(m-\ell)}) \ .$$

Further on:

$$\Pr(\mathbf{Z}^{(n)} = \mathbf{z}^{(n)} | \mathbf{K} = \mathbf{k}, \mathbf{U}^{(m-\ell)} = \mathbf{u}^{(m-\ell)}) = \Pr(\mathbf{V}^{(n)} = \mathbf{v}^{(n)} = \mathbf{z}^{(n)} \oplus f^{(n)}(\mathbf{k}) \oplus [\mathbf{a}^{(\ell)} || \mathbf{u}^{(m-\ell)}]\mathbf{G})$$

and accordingly

$$\Pr(\mathbf{Z}^{(n)} = \mathbf{z}^{(n)} | \mathbf{K} = \mathbf{k}) = \sum_{\mathbf{u}^{(m-\ell)}} \Pr(\mathbf{U}^{(m-\ell)} = \mathbf{u}^{(m-\ell)}) \Pr(\mathbf{V}^{(n)} = \mathbf{v}^{(n)} = \mathbf{z}^{(n)} \oplus f^{(n)}(\mathbf{k}) \oplus [\mathbf{a}^{(\ell)} || \mathbf{u}^{(m-\ell)}]\mathbf{G})$$

$$= \frac{1}{2^{m-\ell}} \sum_{w=0}^{n} \alpha(w) p^w (1-p)^{n-w} \ , \ , \tag{3}$$

where $\alpha(w)$ is the number of different vectors $\mathbf{u}^{(m-\ell)}$ which imply the same $w = Hwt(\mathbf{z}^{(n)} \oplus f^{(n)}(\mathbf{k}) \oplus [\mathbf{a}^{(\ell)} || \mathbf{u}^{(m-\ell)}]\mathbf{G})$.

Accordingly, the following statements can be proved (the proofs are omitted because of the extended abstract length limitation).

Lemma 1. The posterior probability of the secret key assuming the chosen plaintext attack, when the sample $\{\mathbf{z}_t^{(n)}\}_{t=1}^{\tau}$ is available, is given by the following:

$$\Pr(\mathbf{K}=\mathbf{k}|\{\mathbf{Z}_t^{(n)}=\mathbf{z}_t^{(n)}\}_{t=1}^{\tau}) = \frac{\sum_{w=0}^{\tau n}\alpha^*(w)p^w(1-p)^{n-w}}{\sum_{\mathbf{k}}\sum_{w=0}^{\tau n}\alpha^*(w)p^w(1-p)^{n-w}} \ , \tag{4}$$

where $\alpha^*(w)$ is equal to the number of different combinations of the vectors $\{\mathbf{u}_t^{(m-\ell)}\}_{t=1}^{\tau}$ which for given $\{\mathbf{z}_t^{(n)}\}_{t=1}^{\tau}$, $\{\mathbf{a}_t^{(\ell)}\}_{t=1}^{\tau}$ and $\mathbf{k}$ provide that $Hwt(||_{t=1}^{\tau}[\mathbf{z}_t^{(n)}\oplus f_t^{(n)}(\mathbf{k})\oplus[\mathbf{a}_t^{(\ell)}||\mathbf{u}_t^{(m-\ell)}]\mathbf{G}]) = w$, $w\in\{0,1,...,n\tau\}$, $||_{t=1}^{\tau}[\cdot]$ denotes the concatenation of τ n-dimensional vectors, $Hwt(\cdot)$ denotes the Hamming weight of the considered binary vector, and $\sum_{\mathbf{k}}(\cdot)$ denotes summation over all possible keys.

Theorem 1. The equivocation of secret key in the chosen plaintext attack scenario, when the sample $\{\mathbf{z}_t^{(n)}\}_{t=1}^{\tau}$ is available, is given by the following:

$$H(\mathbf{K}|\{\mathbf{Z}_t^{(n)}\}_{t=1}^{\tau}) = 2^{-(|\mathbf{k}|+\tau(m-\ell))}\sum_{\mathbf{z}^{(\tau n)}}(\sum_{\mathbf{k}}\sum_{w=0}^{\tau n}\alpha^*(w)p^w(1-p)^{\tau n-w})\cdot\log_2\sum_{\mathbf{k}}\sum_{w=0}^{\tau n}\alpha^*(w)p^w(1-p)^{\tau n-w}$$

$$-2^{-(|\mathbf{k}|+\tau(m-\ell))}\sum_{\mathbf{z}^{(\tau n)}}\sum_{\mathbf{k}}(\sum_{w=0}^{\tau n}\alpha^*(w)p^w(1-p)^{\tau n-w})\cdot\log_2(\sum_{w=0}^{\tau n}\alpha^*(w)p^w(1-p)^{\tau n-w}) \ .$$

where $|\mathbf{k}|$ is length of the secret key $\mathbf{k}$, $\mathbf{z}^{(\tau n)} = ||_{t=1}^{\tau}\mathbf{z}_t^{(n)}$, $\alpha^*(w)\geq 0$ is equal to the number of different combinations of the vectors $\{\mathbf{u}_t^{(m-\ell)}\}_{t=1}^{\tau}$ which for given $\{\mathbf{z}_t^{(n)}\}_{t=1}^{\tau}$, $\{\mathbf{a}_t^{(\ell)}\}_{t=1}^{\tau}$ and $\mathbf{k}$ provide that $Hwt(||_{t=1}^{\tau}[\mathbf{z}_t^{(n)}\oplus f_t^{(n)}(\mathbf{k})\oplus[\mathbf{a}_t^{(\ell)}||\mathbf{u}_t^{(m-\ell)}]\mathbf{G}]) = w, \quad w\in\{0,1,...,n\tau\}$, $||_{t=1}^{\tau}[\cdot]$ denotes the concatenation of τ n-dimensional vectors, $Hwt(\cdot)$ denotes the Hamming weight of the considered binary vector, and $\sum_{\mathbf{k}}(\cdot)$ denotes summation over all possible keys.

Theorem 2. When $p=0$ we have the following:

$$H(\mathbf{K}|\{\mathbf{Z}_t^{(n)}\}_{t=1}^{\tau}) = \begin{cases} 0 & , \quad |\mathbf{k}|+\tau(m-\ell)\leq\tau n \\ |\mathbf{k}|+\tau(m-\ell)-\tau n \ , & |\mathbf{k}|+\tau(m-\ell)>\tau n \end{cases} ,$$

and

$$H(\mathbf{K}|\{\mathbf{Z}_t^{(n)}\}_{t=1}^{\tau}) \leq min\{|\mathbf{k}|,\tau(m-\ell)\} = \min\{H(\mathbf{K}),H(\mathbf{U}^{(\tau m-\tau\ell)})\} \ ,$$

where $H(\mathbf{K})$ and $H(\mathbf{U}^{(\tau m-\tau\ell)})$ denote the entropies of the keys and the employed randomness, respectively.

$H(\mathbf{K}|\{\mathbf{Z}_t^{(n)}\}_{t=1}^{\tau})$ is a monotony increasing function of the parameter p and it reaches its maximal value equal to $|\mathbf{k}|$ when $p=1/2$.

Theorem 3. When $p<1/2$, $H(\mathbf{K}|\{\mathbf{Z}_t^{(n)}\}_{t=1}^{\tau})$ is a monotony decreasing function of the parameters n and τ such that:

$$H(\mathbf{K}|\{\mathbf{Z}_t^{(n)}\}_{t=1}^{\tau}) < |\mathbf{k}| \qquad \text{and} \qquad \lim_{\tau\to\infty}H(\mathbf{K}|\{\mathbf{Z}_t^{(n)}\}_{t=1}^{\tau}) = 0 \ .$$

4 Computational Complexity Analysis of the Security

The following two statements relate the computational complexity based security of the considered model to the hardness of decoding a random linear block code which is known as NP-hard problem as shown in [2], and the average hardness of the LPN problem considered in [4], [5] and [8], for example. (The following Theorems 3 and 4 yield simplified/informal statements because of the space limitation.)

Theorem 4. The complexity of recovering the secret key $\mathbf{k}$ in the chosen plaintext attack based on the algebraic representation of the considered stream cipher is lower bounded by the complexity of decoding a random linear block code after a binary symmetric channel with the crossover probability equal to $\epsilon = \frac{1-(1-2p)^{(m-\ell)/2}}{2}$.

The following statement shows that the problem of distinguishing to which of two possible plaintexts corresponds the cipherthet generated by the considered stream cipher is as hard as solving certain LPN problem. More formally, let $\mathbf{a}_1$ and $\mathbf{a}_2$ be two chosen plaintext known to an attacker, and let one of them has been randomly selected and encrypted by the considered stream cipher. In the case of the indistinguishability (IND) security evaluation, the goal is to evaluate the advantage, in comparison with the random guessing (which provides the success probability equal to 1/2), of an attacker to determine wether $\mathbf{a}_1$ or $\mathbf{a}_2$ has been encrypted.

Theorem 5. Let the considered randomized stream cipher employs the keystream generator which is a linear finite state machine and the cipher parameters are (k, ℓ, m, n, p). Assumption that there is an adversary $\mathcal{A}$, running in time T, and attacking the cipher in the sense of distinguishing under chosen plaintext attack with advantage δ by making at most q queries to the encryption oracle implies that there is an algorithm $\mathcal{L}$ such that making $O(q)$ oracle queries and running in time $O(T)$ can solve certain LPN problem corresponding to the noise parameter $\epsilon = \frac{1-(1-2p)^{(m-\ell/2}}{2}$.

References

1. B. Applebaum, D. Cash, C. Peikert and A. Sahai, ”Fast Cryptographic Primitives and Circular-Secure Encryption Based on Hard Learning Problems”, CRYPTO 2009, *Lecture Notes in Computer Science*, vol. 5677, pp. 595-618, Aug. 2009.
2. E.R. Berlekamp, R.J. McEliece, and H.C.A. van Tilborg, “On the Inherent Intractability of Certain Coding Problems”, *IEEE Trans. Info. Theory*, vol. 24, pp. 384-386, 1978.
3. A. Blum, M. Furst, M. Kearns and R. Lipton, “Cryptographic Primitives Based on Hard Learning Problems”, CRYPTO 1993, *Lecture Notes in Computer Science*, vol. 773, pp. 278291, 1994.
4. A. Blum, A. Kalai and H. Wasserman, “Noise-Tolerant Learning, the Parity Problem, and the Statistical Query Model”, *Journal of the ACM*, vol. 50, no. 4, pp. 506-519, July 2003.
5. M. Fossorier, M.J. Mihaljević, H. Imai, Y. Cui and K. Matsuura, ”An Algorithm for Solving the LPN Problem and its Application to Security Evaluation of the HB Protocols for RFID Authentication”, INDOCRYPT 2006, *Lecture Notes in Computer Science*, vol. 4329, pp. 48-62, Dec. 2006.
6. H. Gilbert, M.J.B. Robshaw, and Y. Seurin, ”How to Encrypt with the LPN Problem”, ICALP 2008, Part II, *Lecture Notes in Computer Science*, vol. 5126, pp. 679-690, 2008.
7. H.N. Jendal, Y.J.B. Kuhn, and J.L. Massey, “An information-theoretic treatment of homophonic substitution”, EUROCRYPT’89, *Lecture Notes in Computer Science*, vol. 434, pp. 382-394, 1990.
8. E. Levieil and P.-A. Fouque, ”An Improved LPN Algorithm”, SCN 2006, *Lecture Notes in Computer Science*, vol. 4116, pp. 348-359, 2006.
9. J. Massey, “Some Applications of Source Coding in Cryptography”, *European Transactions on Telecommunications*, vol. 5, pp. 421-429, July-August 1994.
10. M.J. Mihaljević and H. Imai, “An approach for stream ciphers design based on joint computing over random and secret data”, *Computing*, vol. 85, no. 1-2, pp. 153-168, June 2009.
11. M.J. Mihaljević, ”A Framework for Stream Ciphers Based on Pseudorandomness, Randomness and Error-Correcting Coding”, in *Enhancing Cryptographic Primitives with Techniques from Error Correcting Codes*, Editors B. Preneel, S. Dodunekov, V. Rijmen and S. Nikova, Vol. 23 in the *NATO Science for Peace and Security Series - D: Information and Communication Security*, pp. 117-139, IOS Press, Amsterdam, The Netherlands, June 2009.
12. M.J. Mihaljević and H. Imai, ”A Security Evaluation of Certain Stream Ciphers which Involve Randomness and Coding”, *2010 Int. Symp. on Inform. Theory and its Appl. - ISITA 2010*, Taichung, Taiwan, Oct. 17-20, 2010, Proceedings, pp. 789-794, IEEE, 2010.
13. R. Rivest and T. Sherman, ”Randomized Encryption Techniques”, *Advances in Cryptology: Proceedings of CRYPTO ’82*, Plemum, New Yourk, pp. 145-163, 1983.

The Cryptographic Power of Random Selection

Matthias Krause and Matthias Hamann

Theoretical Computer Science
University of Mannheim
Mannheim, Germany

Abstract. The principle of random selection and the principle of adding biased noise are new paradigms used in several recent papers for constructing lightweight RFID authentication protocols. The cryptographic power of adding biased noise can be characterized by the hardness of the intensively studied Learning Parity with Noise (LPN) Problem. In analogy to this, we identify a corresponding learning problem called *RandomSelect* for random selection and study its complexity. Given L secret linear functions $f_1, \ldots, f_L : \{0,1\}^n \longrightarrow \{0,1\}^a$, $\mathit{RandomSelect}\,(L, n, a)$ denotes the problem of learning $f_1, \ldots, f_L$ from values $(u, f_l\,(u))$, where the secret indices $l \in \{1, \ldots, L\}$ and the inputs $u \in \{0,1\}^n$ are randomly chosen by an oracle. We take an algebraic attack approach to design a nontrivial learning algorithm for this problem, where the running time is dominated by the time needed to solve full-rank systems of linear equations over $O\left(n^L\right)$ unknowns. In addition to the mathematical findings relating correctness and average running time of the suggested algorithm, we also provide an experimental assessment of our results.

Keywords: Lightweight Cryptography, Algebraic Attacks, Algorithmic Learning, Foundations and Complexity Theory

1 Introduction

The very limited computational resources available in technical devices like RFID (radio frequency identification) tags implied an intensive search for lightweight authentication protocols in recent years. Standard block encryption functions like Triple-DES or AES seem to be not suited for such protocols largely because the amount of hardware to implement and the energy consumption to perform these operations is too high (see, e.g., [7] or [13] for more information on this topic).

This situation initiated two lines of research. The first resulted in proposals for new lightweight block encryption functions like PRESENT [4], KATAN and KTANTAN [10] by use of which standard block cipher-based authentication protocols can be made lightweight, too. A second line, and this line we follow in the paper, is to look for new cryptographic paradigms which allow for designing new symmetric lightweight authentication protocols. The two main suggestions discussed so far in the relevant literature are the principle of random selection and the principle of adding biased noise.

The principle of adding biased noise to the output of a linear basis function underlies the HB-protocol of Juels and Weis [13] as well as its variants HB$^+$, HB$^\#$, and Trusted-HB (see [13], [11], and [6], respectively). The protocols of the HB-family are provably secure against passive attacks with respect to the Learning Parity with Noise Conjecture but the problem to design HB-like protocols which are secure against active adversaries seems to be still unsolved.

The principle of random selection underlies, e.g., the CKK-protocols of Cichoń, Klonowski, and Kutyłowski [7] as well as the F_f-protocols in [3] and the Linear Protocols in [14]. It can be described as follows.

Suppose that the verifier Alice and the prover Bob run a challenge-response authentication protocol which uses a lightweight symmetric encryption operation $E : \{0,1\}^n \times \mathcal{K} \longrightarrow \{0,1\}^m$ of block length n, where $\mathcal{K}$ denotes an appropriate key space. Suppose further that E is weak in the sense that a passive adversary can efficiently compute the secret key $K \in \mathcal{K}$ from samples of the form $(u, E_K(u))$. This is obviously the case if E is linear.

Random selection denotes a method for compensating the weakness of E by using the following mode of operation. Instead of holding a single $K \in \mathcal{K}$, Alice and Bob share a collection

$K_1, \ldots, K_L$ of keys from $\mathcal{K}$ as their common secret information, where $L > 1$ is a small constant. Upon receiving a challenge $u \in \{0,1\}^n$ from Alice, Bob chooses a random index $l \in \{1, \ldots, L\}$ and outputs the response $y = E(u, K_l)$. The verification of y with respect to u can be efficiently done by computing $E_{K_l}^{-1}(y)$ for all $l = 1, \ldots, L$.

The main problem our paper is devoted to is to determine the level of security which can be reached by applying this principle of random selection.

Note that the protocols introduced in [7], [3], and [14] are based on random selection of $GF(2)$-linear functions. The choice of linear basis functions is motivated by the fact that they can be implemented efficiently in hardware and have desirable pseudo-random properties with respect to a wide range of important statistical tests.

It is quite obvious that, with respect to passive adversaries, the security of protocols which use random selection of linear functions can be bounded from above by the complexity of the following learning problem referred to as $RandomSelect\,(L, n, a)$: Learn $GF(2)$-linear functions $f_1, \ldots, f_L : \{0,1\}^n \longrightarrow \{0,1\}^a$ from values $(u, f_l\,(u))$, where the secret indices $l \in \{1, \ldots, L\}$ and the inputs $u \in \{0,1\}^n$ are randomly chosen by an oracle. In order to illustrate this, we will sketch how an efficient learning algorithm for $RandomSelect\,(L, n, a)$ can be used for attacking the linear $(n, k, L)^+$-protocol described by Krause and Stegemann [14].

Consequently, in the full version of this paper, we present an algebraic attack approach for solving $RandomSelect(L, n, a)$. The running time of our algorithm is dominated by the effort necessary to solve a full-rank system of linear equations of $O(n^L)$ unknowns over the field $GF(2^a)$. Note that trivial approaches for solving the problem $RandomSelect\,(L, n, a)$ lead to a running time exponential in n.

In recent years, people from cryptography as well as from complexity and coding theory devoted much interest to the solution of learning problems around linear structures. Prominent examples in the context of lightweight cryptography are the works by Goldreich and Levin [12], Regev [16], and Arora and Ge [2]. But all these results are rather connected to the Learning Parity with Noise Problem. To the best of our knowledge, there are currently no nontrivial results with respect to the particular problem of learning randomly selected linear functions, which is studied in our present paper.

We are strongly convinced that the complexity of $RandomSelect$ also defines a lower bound on the security achievable by protocols using random selection of linear functions, e.g., the improved $(n, k, L)^{++}$-protocol in [14]. Thus, the running time of our algorithm hints at how the parameters n, k, and L should be chosen in order to achieve an acceptable level of cryptographic security. Note that choosing $n = 128$ and $L = 8$ or $n = 256$ and $L = 4$, solving $RandomSelect\,(L, n, a)$ by means of our algorithm implies solving a system of around 2^{28} unknowns, which should be classified as sufficiently difficult in many practical situations.

The full version of this paper is organized as follows. In sections 2, 3, and 4, our learning algorithm, which conducts an algebraic attack in the spirit of [17], will be described in full detail. We represent the L linear basis functions as assignments A to a collection $X = \left(x_i^l\right)_{i=1,\ldots,n, l=1,\ldots,L}$ of variables taking values from the field $K = GF(2^a)$. We will then see that each example $(u, f_l\,(u))$ induces a degree-L equation of a certain type in the X-variables, which allows for reducing the learning problem $RandomSelect\,(L, n, a)$ to the problem of solving a system of degree-L equations over K. While, in general, the latter problem is known to be NP-hard, we can show an efficient way to solve this special kind of systems.

One specific problem of our approach is that, due to inherent symmetries of the degree-L equations, we can never reach a system which has full linear rank with respect to the corresponding monomials. In fact, this is the main difference between our learning algorithm and the well-known algebraic attack approaches for cryptanalyzing LFSR-based keystream generators (see, e.g., [15], [8], [9], [1]).

We circumvent this problem by identifying an appropriate set $T(n, L)$ of basis polynomials of degree at most L which allow to express the degree-L equations as linear equations over $T(n, L)$.

The choice of $T(n, L)$ will be justified by a fundamental Theorem saying that if $|K| \geq L$, then the system of linear equations over $T(n, L)$ induced by all possible examples has full rank $|T(n, L)|$. (Note that according to another Theorem, this is not true if $|K| < L$.) Our experiments, which are presented in section 5 of the full paper, indicate that if $|K| \geq L$, then with probability close to one, the number of examples needed to get a full rank system over $T(n, L)$ exceeds $|T(n, L)|$ only by a small constant factor. This implies that the effort to compute the unique *weak* solution $t(A) = (t_*(A))_{t_* \in T(n,L)}$ corresponding to the *strong* solution A equals the time needed to solve a system of $|T(n, L)|$ linear equations over K.

But in contrast to the algebraic attacks in [15], [8], [9], [1], we still have to solve another nontrivial problem, namely, to compute the *strong* solution A, which identifies the secret functions $f_1, \ldots, f_L$, from the unique weak solution. An efficient way to do this will complete our learning algorithm for $RandomSelect\,(L, n, a)$ in section 4 of the full paper. Finally, we also provide an experimental evaluation of our estimates using the computer algebra system Magma [5] in section 5 and conclude with a discussion of the obtained results as well as an outlook on potentially fruitful future work in section 6.

References

1. F. Armknecht and M. Krause. Algebraic attacks on combiners with memory. In *Proceedings of Crypto 2003*, volume 2729 of *LNCS*, pages 162–176. Springer, 2003.
2. S. Arora and R. Ge. New algorithms for learning in presence of errors. Submitted, 2010. `http://www.cs.princeton.edu/~rongge/LPSN.pdf`.
3. E.-O. Blass, A. Kurmus, R. Molva, G. Noubir, and A. Shikfa. The F_f-family of protocols for RFID-privacy and authentication. In *5th Workshop on RFID Security, RFIDSec'09*, 2009.
4. A. Bogdanov, L. R. Knudsen, G. Leander, C. Paar, A. Poschmann, M. J. B. Robshaw, Y. Seurin, and C. H. Vikkelsoe. PRESENT: An ultra-lightweight block cipher. In *Proceedings of Cryptographic Hardware and Embedded Systems (CHES) 2007*, volume 4727 of *LNCS*, pages 450–466. Springer, 2007.
5. W. Bosma, J. Cannon, and C. Playoust. The Magma algebra system. I. The user language. *J. Symbolic Comput.*, 24(3-4):235–265, 1997.
6. J. Bringer and H. Chabanne. Trusted-HB: A low cost version of HB$^+$ secure against a man-in-the-middle attack. *IEEE Trans. Inform. Theor.*, 54:4339–4342, 2008.
7. J. Cichoń, M. Klonowski, and M. Kutyłowski. Privacy protection for RFID with hidden subset identifiers. In *Proceedings of Pervasive 2008*, volume 5013 of *LNCS*, pages 298–314. Springer, 2008.
8. N. Courtois. Fast algebraic attacks on stream ciphers with linear feedback. In *Proceedings of Crypto 2003*, volume 2729 of *LNCS*, pages 176–194. Springer, 2003.
9. N. Courtois and W. Meier. Algebraic attacks on stream ciphers with linear feedback. In *Proceedings of Eurocrypt 2003*, volume 2656 of *LNCS*, pages 345–359. Springer, 2003.
10. C. De Cannière, O. Dunkelman, and M. Knežević. KATAN and KTANTAN – A family of small and efficient hardware-oriented block ciphers. In *Proceedings of the 11th International Workshop on Cryptographic Hardware and Embedded Systems (CHES) 2009*, volume 5747 of *LNCS*, pages 272–288. Springer, 2009.
11. H. Gilbert, M. J. B. Robshaw, and Y. Seurin. HB$^\#$: Increasing the security and efficiency of HB$^+$. In *Proceedings of Eurocrypt 2008*, volume 4965 of *LNCS*, pages 361–378, 2008.
12. O. Goldreich and L. A. Levin. A hard-core predicate for all one-way functions. In *Proceedings of the twenty-first annual ACM symposium on Theory of computing (STOC)*, pages 25–32. ACM Press, 1989.
13. A. Juels and S. A. Weis. Authenticating pervasive devices with human protocols. In *Proceedings of Crypto 2005*, volume 3621 of *LNCS*, pages 293–308. Springer, 2005.
14. M. Krause and D. Stegemann. More on the security of linear RFID authentication protocols. In *Proceedings of SAC 2009*, volume 5867 of *LNCS*, pages 182–196. Springer, 2009.
15. W. Meier, E. Pasalic, and C. Carlet. Algebraic attacks and decomposition of boolean functions. In *Proceedings of Eurocrypt 2004*, volume 3027 of *LNCS*, pages 474–491. Springer, 2004.
16. O. Regev. On lattices, learning with errors, random linear codes, and cryptography. In *Proceedings of the thirty-seventh annual ACM symposium on Theory of computing (STOC)*, pages 84–93. ACM Press, 2005.
17. A. Shamir, J. Patarin, N. Courtois, and A. Klimov. Efficient algorithms for solving overdefined systems of multivariate polynomial equations. In *Proceedings of Eurocrypt 2000*, volume 1807 of *LNCS*, pages 474–491. Springer, 2000.

Fast Parallel Keyed Hash Functions Based on Chaotic Maps (PKHC)

Mahmoud M. Maqableh [a,b]

[a] School of Engineering and Computer Sciences, Durham University, South Road, Durham, DH1 3LE, United Kingdom.

[b] Faculty of Business, Management Information Systems Department, Jordan University, Amman 11942, Jordan.

[*] Correspondence author. Tel.:+44(0)19 3341700; fax: +44 (0)1913341701. E-mail addresses: m.m.maqableh@dur.ac.uk, maqableh@ju.edu.jo

Abstract:

In the last decade, various hash functions based on chaotic maps were proposed. Many of the proposed algorithms are proved as unsecure or slaw speed hash function algorithms. In this paper, we propose a novel parallel chaotic keyed hash functions. The input message is partitioned into fixed length of blocks. Hash round function is processed all message blocks and generates intermediate hash value. The chaotic map is used to produce the final hash value by mixing all rounds intermediate hash values. Hash round functions are implemented by one of the chaotic maps and can work in parallel mode to provide high performance and security. The proposed hash function has a very simple and flexible design, which produces different lengths of keyed hash functions using different chaotic maps. The theoretical analyses and computer simulations confirmed that the proposed hash function satisfies the requirements of cryptographic hash function with high security and speed. By comparing the proposed hash function with several existing hash functions, we conclude the proposed hash function provides higher flexibility, better performance, and higher security than many other existing hash functions. These properties make it suitable for different applications and protocols, such as Secure Socket Layer (SSL), Transport Layer Security (TLS) and e-commerce applications.

Block Ciphers Based on Wavelet Decomposition of Splines

Alla Levina

This paper presents the new idea which can be applied in the secret-key cryptosystems. The idea is based on using the theory of wavelet decomposition of splines.

Proposed paper discusses a new class of algorithms obtained on a theory of the spline-wavelet decompositions on a nonuniform sets. Theory of wavelet decomposition of splines has been used before to process discreet signals but never in cryptography.

Our proposal to create cryptoalgoritms which will use only mathematical calculation, the algorithm that can processes data blocks up to 2048 bits and more. These researches were made for splines of the first, second and third degree. Algorithms based on splines of third degree works slower but they stay stronger to different cryptoattacks. It also can be made combination of splines of different degrees in one algorithm.

Theory of wavelet-decomposition of splines for creation of block ciphers can be applied in different ways.

The presented algorithms do not have XOR operation with the round key and they do not use S-boxes. Diffusion over multiple rounds we get by mathematical functions.

As a minuses of this algorithm we can mention what not all the bytes are getting encipher on each round, some of them just getting moved on several positions, but not like in Feistel Structure.

Now we will shortly give the idea of wavelet-decomposition of splines.

On the primary set X we will build splines ω_i. Set X consists from the elements $\{x_i\}_{i=0,\dots,L-1}$, where $\{x_i\}_{i=0,\dots,L-1}$ natural numbers. L is a number of elements in the set X.

For wavelet decomposition of splines we take out one element x_k from our primary set X and we get new set $\overline{X}$.

On the new set $\overline{X}$ we can build new splines $\overline{\omega}_j$ but these new splines can be represent as a combination of splines which were build before on the set X.

Also splines $\omega_j(t)$ can be gotten with the help of new splines $\overline{\omega}_j(t)$, it helps us to restore information.

This idea gives us two types of formulas: formulas of decomposition and formulas of reconstruction. Step by step we take out elements from our primary set X and build splines which use new set (in this realization each time we take out just one element and we get new set and new splines, in an other realizations it we take few elements each time).

If we have information stream c_i and we want to get the new stream $\overline{c_i}$ based on set $\overline{X}$ we will use formulas of decomposition. Formulas of reconstruction will help us to restore stream c_i, using stream $\overline{c_i}$.

A process of enciphering and deciphering consists of K identical rounds.

The number of rounds is denoted by K, $\mathbb{K}_{X\gamma}$ is a key length, M is a block length (in the table below M and $\mathbb{K}_{X\gamma}$ are bytes).

Let $\mathbb{K} = (X, \gamma)$ be *a key*; here X is an ordered set, $X = \{x_j\}_{j=0,\dots,L-1}$, where L is number of elements in the set X and γ is the order of ejection of elements from the set. The key consist from two sets.

Number of rounds and key length as a function of the block length given in Table 1, but it can be changed.

	K	$\mathbb{K}_{X\gamma}$
$M = 8$ bytes	6	15
$M = 16$ bytes	14	31
$M = 24$ bytes	22	47
$M = 32$ bytes	30	63
$M = 64$ bytes	62	127
$M = 128$ bytes	126	255
$M = 256$ bytes	254	511

Table 1.

A sequence $C = \{c_i\}_{i=0,\dots,M-1}$ *is a plaintext;* $|C| = M$ is a quantity of elements which are ciphered, C is the ordered set.

Elements $\{c_i\}_{i=0,\dots,M-1}$ and $\{x_j\}_{j=0,\dots,L-1}$ are bytes (we are working with one-bytes words, but we also can work with 4-bytes words).

Set X and C can be periodic with the period T so $x_j = x_{j+T}$ and $c_i = c_{i+T} \quad \forall j \in \mathbb{Z}$.

Process of enciphering bases on the formulas of decomposition from wavelet theory, after K rounds we obtain the ciphertext. For deciphering we use formulas of reconstruction.

Process of enciphering and deciphering consists from two steps, creation of round key and round transformation.

All calculations goes in finite fields by prime polynomial N, in our realization we toke polynomial from algorithm Rijndael $x^8 + x^4 + x^3 + x + 1$.

New Universal Hash Functions

Aysajan Abidin

Department of Electrical Engineering,
Linköping University, SE-581 83 Linköping, Sweden
aysajan@isy.liu.se

Universal hash functions were introduced by Wegman and Carter in 1979, and since then they have been extensively studied. They are used in diverse cryptographic tasks such as unconditionally secure authentication, error-correction and privacy amplification (or randomness extraction). Various constructions of Universal hash function classes by Wegman and Carter, Stinson, Chaum et al., and den Boer are known. All have different description lengths and differ in terms of their computational efficiencies.

This presentation addresses a new construction of Universal hash functions. In particular, a new construction of ϵ-Almost Strongly Universal (ϵ-ASU$_2$) hash functions is presented, using the idea of LFSR-based hashing by Krawczyk. This new construction is very efficient and requires much smaller description than the well-known Wegman-Carter construction of ϵ-ASU$_2$ hash functions.

Before looking at the new construction, let us recall the definitions of Universal and ϵ-ASU$_2$ hash functions and the composition theorem for Universal hash functions.

Definition 1 (Universal$_2$ hash functions). *Let $\mathcal{M}$ and $\mathcal{T}$ be finite sets. A class $\mathcal{H}$ of hash functions from $\mathcal{M}$ to $\mathcal{T}$ is* Universal$_2$ *if there exist at most $|\mathcal{H}|/|\mathcal{T}|$ hash functions $h \in \mathcal{H}$ such that $h(m_1) = h(m_2)$ for any two distinct $m_1, m_2 \in \mathcal{M}$.*

If there are at most $\epsilon|\mathcal{H}|$ hash functions instead, it is called ϵ-almost universal$_2$ (ϵ-AU_2).

Definition 2 (ϵ-ASU$_2$ hash functions). *Let $\mathcal{M}$ and $\mathcal{T}$ be as before. A class $\mathcal{H}$ of hash functions from $\mathcal{M}$ to $\mathcal{T}$ is ϵ-ASU_2 if the following two conditions are satisfied:*

(a) *The number of hash functions in $\mathcal{H}$ that takes an arbitrary $m_1 \in \mathcal{M}$ to an arbitrary $t_1 \in \mathcal{T}$ is exactly $|\mathcal{H}|/|\mathcal{T}|$.*

(b) *The fraction of those functions that also takes an arbitrary $m_2 \neq m_1$ in $\mathcal{M}$ to an arbitrary $t_2 \in \mathcal{T}$ (possibly equal to t_1) is at most ϵ.*

If $\epsilon = 1/|\mathcal{T}|$, then $\mathcal{H}$ is called SU_2.

Theorem 1 (Composition). *Let $\mathcal{F}$ be a set of ϵ_1-AU_2 hash functions from $\mathcal{M} \to \mathcal{Z}$, and let $\mathcal{H}$ be a set of SU_2 hash functions from $\mathcal{Z} \to \mathcal{T}$. Then, $\mathcal{G} = \mathcal{H} \circ \mathcal{F}$ is an ϵ-ASU_2 hash function family from $\mathcal{M} \to \mathcal{T}$ with $\epsilon = \epsilon_1 + 1/|\mathcal{T}|$.*

The construction is as follows: Our aim is to construct an $2/|\mathcal{T}|$-ASU$_2$ family with smaller description than the Wegman-Carter construction, which requires a key of length $(\log|\mathcal{T}| + \log\log\log|\mathcal{M}|)4\log|\mathcal{M}|$ to describe a hash function in the family, here the log stands for the binary logarithm. To this end, we use the composition theorem above for constructing such a hash function family. By composing an LFSR-based $2\log|\mathcal{M}|/|\mathcal{Z}|$-AU$_2$ hash functions from $\mathcal{M} \to \mathcal{Z}$ with an SU$_2$ hash functions from $\mathcal{Z} \to \mathcal{T}$, we obtain a $2\log|\mathcal{M}|/|\mathcal{Z}| + 1/|\mathcal{T}|$-ASU$_2$ hash functions. To make $2\log|\mathcal{M}|/|\mathcal{Z}| + 1/|\mathcal{T}| = 2/|\mathcal{T}|$, we let $|\mathcal{Z}|$ be equal to $2\log|\mathcal{M}||\mathcal{T}|$. For the LFSR-based construction, the required hash function description is $2\log|\mathcal{Z}| + 1$. But for the SU$_2$ hash functions the description length is roughly $2\log|\mathcal{Z}|$. Therefore, for this construction of $2/|\mathcal{T}|$-ASU$_2$ hash functions, the required description length $4\log|\mathcal{Z}| + 1 = 5 + 4(\log|\mathcal{T}| + \log\log|\mathcal{M}|)$, which is much shorter than $(\log|\mathcal{T}| + \log\log\log|\mathcal{M}|)4\log|\mathcal{M}|$. For instance, for $\log|\mathcal{M}| = 2^{30}$ and $\log|\mathcal{T}| = 64$, the above presented construction requires 280 bits of key to achieve $2/2^{64}$-ASU$_2$, while the Wegman-Carter approach requires 8268 bits to achieve $12/2^{64}$-ASU$_2$.

This new construction first uses LFSR, which can be efficiently be implemented in both hardware and/or software, to map the big number (the long message) to a much smaller number (the intermediate string). Then, the intermediate short string is mapped by an SU$_2$ hash function to a tag. Therefore, this new construction is computationally very efficient.

Cryptanalysis of the Light-Weight Cipher A2U2 – Extended Abstract

Mohamed Ahmed Abdelraheem, Julia Borghoff, Erik Zenner

Technical University of Denmark, DK-2800 Kgs. Lyngby, Denmark
{M.A.Abdelraheem,J.Borghoff,E.Zenner}@mat.dtu.dk

Introduction

Increasingly, small computing devices become a part of the pervasive communication infrastructure. Radio Frequency Identification (RFID) technology is a corner stone of ubiquitous computing. RFID tags are low cost devices which communicate wirelessly with a reader and are used for the purpose of identification and tracking. Using RFID technology it is possible, e.g., to identify products in a warehouse with a unique identification number. Therefore RFID tags are expected to replace bar codes in the future. This becomes even more likely with the invention of IC-printing (integrated circuit printing) [4] which makes the production of RFID tags even cheaper.

With the increasing deployment of low cost computing devices there comes also a demand for security solutions. As we are faced with extremely resource constrained environments w.r.t. power consumption and area, traditional cryptographic algorithms cannot be employed. Thus, there is a need for specially tailored encryption algorithms.

At IEEE RFID 2011, David et al. proposed a new cryptographic primitive for use with RFID [2]. The design is a stream cipher called A2U2 and was inspired by the lightweight block cipher KATAN [3]. A2U2 can be implemented using 284GE and has an output rate of 1 bit per cycle.

Shortly after its publication, a **chosen-plaintext attack** was published on IACR Eprint by Chai et al. [1], claiming to break the cipher using extremely few computational resources. As it turns out, however, this attack is not applicable since it works with an erroneous description of the cipher. In this work, we show why the attack does not work and how it can be repaired.

We then continue by describing attacks for a **known-plaintext scenario**. Firstly, we propose a guess-and-determine attack that is faster than exhaustive search, requiring about 2^{49} guesses.

In addition, we analyze the key/IV setup. A special design feature of A2U2 is that the number of initialization rounds varies from 9 to 126, depending on a part of the key called counter key. We propose a differential-style attack that enables us to find this counter key. Moreover, we present an attack with complexity 2^{38} that recovers the master key in the case where only 9 initialization rounds are used. Both attacks are **chosen-IV attacks**, i.e. they require the attacker to choose the initialization vector.

The Stream Cipher A2U2

The cipher's inner state consists of a counter LFSR C (7 bit), two non-linear feedback shift registers (NFSRs) A and B (17 and 9 bit), and a key register K (56 bit). Thus, the total inner state size is 89 bit.

The two NFSRs are inspired by KATAN [3], where the feedback function of the first register provides the feedback of the second register and vice versa. Thus, the update of NFSR B depends only on bits of register A, while the update of register A uses bits of B and a derived value h_t. This h_t is determined by five consecutive bits of the master key, the counter LFSR C and one bit from NFSR A and can be presented as a quadratic polynomial. Afterwards the next five key bits are processed.

In order to generate the ciphertext, the cipher deploys a form for irregular output mechanism; it outputs either encrypted plaintext bits or pseudo-random bits depending on the content of NFSR cell A_t. Plaintext bits have to "wait" until $A_t = 1$ before being encrypted. If we denote the plaintext string by $P = (P_0, P_1, \ldots)$ and if we define $\sigma(t) = \sum_{i=0}^{t-1} A_t$ with $\sigma(0) = 0$, then the output of the cipher in round t is:

$$Y_t = \mathrm{MUX}_{A_t}(B_t + C_t, B_t + P_{\sigma(t)}),$$

where $\mathrm{MUX}_x(y, z) = y$ if $x = 0$ and $\mathrm{MUX}_x(y, z) = z$ otherwise.

The cipher's 61-bit key is split into two parts: The master and the counter key. The registers A and B are initialized with 26 bits of the master key xored with part of the IV. The counter LFSR C is initialized using the 5-bit counter key and the remaining IV bits. The cipher runs for a varying number of initialization rounds depending on the counter key. It is important to note that the counter register contains all ones at the beginning of the encryption process; thus, the counter value is known at any time during the encryption.

A Chosen-Plaintext Attack

In [1], a very efficient chosen plaintext attack against A2U2 is proposed. However, the attack contains a flaw that makes it unapplicable against the real A2U2 cipher. The wrong assumption is that the plaintext bits are used at rate of 1 bit per ciphertext bit. Thus, choosing a plaintext which is the complement of the counter sequence would allow the attacker to recover the sequence A_t that determines whether a plaintext or a counter bit is encrypted. However, it is not possible to choose a plaintext bit for every round because some plaintext bits are used over several rounds and the attacker does not know in which rounds a plaintext bit will be used.

Fixing this problem, choosing two plaintexts, namely the all-zero and the all-one plaintext, enables us to recover first the sequence B_t and then to determine the positions of the ciphertext where a plaintext bit was used. When we know the sequences A_t and B_t we can generate a linear equation system in the key bits and recover the 56 bits of the master key.

To recover B_t we first consider the ciphertext corresponding to the all-zero plaintext. We know that for all time slots t with $C_t = 0$ it holds that $C_t = P_{\sigma(t)}$ and thus $B_t = Y_t$. We learn about half of the bits of the sequence B. The remaining bits can be learned by repeating the same exercise with the all-one plaintext.

We can use this new information to learn the sequence A_t as well. For every time slot, we pick the ciphertext bit Y_t corresponding to the plaintext bit $P_{\sigma(t)} \neq C_t$. If it holds that $B_t = Y_t + C_t$, then $A_t = 0$, otherwise $A_t = 1$.

Guess-and-Determine Attack

This attack is based on the fact that the derived value h_t, which is used to update register A, can be presented as an at most quadratic polynomial in the master key bits. The idea is to determine h_t for sufficiently many time slots t in order to set up and solve an equation system in the key bits.

We guess the sequence A_t and determine the corresponding value B_t. After 9 guesses, the full register B is known and we can determine the derived value h_t for all further guesses. After 8 additional guesses we also know the full register A; thus we can determine B_t in the following rounds. The knowledge of B_t enables us to determine A_t for time slots t where the counter bit C_t differs from the plaintext bit. This significantly reduces the complexity of the attack to about 2^{49} guesses.

Targeting the low number of initialization rounds

The number of initialization rounds varies from 9 to 126 and is determined by the counter. We propose a differential-style chosen-IV attack that identifies cases where only 9 rounds of initialization were used. For each of the 32 possible counter values we encrypted 1 bit of plaintext under 2^9 state pairs (a certain sparse difference is introduced in the NFSRs). Then we can observe a bias in the ciphertext bit for the 2^9 pairs where only 9 initialization rounds have been used.

When only 9 initialization rounds are applied we can recover 32 master key bits and 6 bits of type h_t with a complexity of 2^{38} using 8 plaintext/ciphertext pairs of length 5 bit. Depending on the counter value C_{t-1} this is done by either guessing derived bits h_t or master key bits in each round. The remaining master key bits can be recovered by brute force.

References

1. Q. Chai, X. Fan, and G. Gong. An ultra-efficient key recovery attack on the lightweight stream cipher A2U2. http://eprint.iacr.org/2011/247, 2011. Version published: 20110518:133751 (posted 18-May-2011 13:37:51 UTC).
2. M. David, D.C. Ranasinghe, and T. Larsen. A2U2: A stream cipher for printed electronics RFID tags. In *Proceedings of IEEE RFID 2011*, pages 240–247, 2011. to appear.
3. C. de Canniere, O. Dunkelman, and Knezevic M. KATAN and KTANTAN - a family of small and efficient hardware-oriented block ciphers. In *CHES 2009*, volume 5747 of *LNCS*, pages 272 – 288, 2009.
4. PolyIC. Information available via, http://www.polyIC.com.

Cryptanalysis of TWIS Block Cipher

Onur Koçak and Neşe Öztop

Institute of Applied Mathematics, Middle East Technical University, Turkey
{onur.kocak,noztop}@metu.edu.tr

Abstract. TWIS is a 128-bit lightweight block cipher that is proposed by Ojha et al. In this work, we analyze the security of the cipher against differential and impossible differential attacks. For the differential case, we mount a full-round attack on TWIS and recover 12 bits of the 32-bit final subkey with 2^{21} complexity. For the other case, we present distinguisher which can be extended to key recovery attack. Also, we showed that the security of the cipher is only 54 bits instead of claimed 128 bits. Moreover, we introduce some observations that compromise the security of the cipher.

Keywords: TWIS, Lightweight Block Cipher, Differential Cryptanalysis, Impossible Differential Distinguisher.

1 Introduction

TWIS is a 128-bit block cipher designed to be used in ubiquitous devices. The cipher, which is inspired from CLEFIA[1], is a 2-branch generalized Feistel Network of 10 rounds. There is no key recovery attack on this cipher up to the authors knowledge. The only analysis is done by Su et al.[2] in which n-round iterative differential distinguishers are presented. However, as the probability of the iterative distinguishers are 1, they cannot be extended to a differential attack to get information about the key.

In this paper, we analyze the security of TWIS block cipher against differential and impossible differential cryptanalysis. We mount a differential attack on full-round TWIS and recover 12 bits of the 32-bit final subkey with a complexity of 2^{21}. This is the first experimental result on TWIS. Also, we present a 9.5-round impossible distinguisher which can be extended to a key recovery attack. Furthermore, by making observations on the key schedule, we show that the cipher offers at most 54-bit security instead of claimed 128-bit. Besides, we mention the potential weaknesses due to the use of subkeys during the encryption and the choice of whitening subkeys. The paper is organized as follows. In Section 2, a 10-round differential attack is presented. Impossible differential distinguisher is proposed in Section 3. Some observations on the algorithm are given in Section 4. Finally Section 5 concludes the paper. The detailed description of TWIS can be found in [3].

2 Differential Attack on TWIS

In this section, we propose a key recovery attack on 10-round TWIS excluding the final key whitening. Our attack is based on a 9.5-round differential distinguisher which is explained in the following section.

2.1 9.5-round Differential Characteristic

The inputs of the F-function are the 1^{st} and the 3^{rd} 32-bit words of the data which are interchanging in the swap operation. There is no rotation operation applied on the 3^{rd} word and the rotation on the 1^{st} word is a 1-bit right rotation. Therefore, if we have 80000000_x as input difference in the 3^{rd} word, this difference will produce zero differences after the F-function with probability 1 during the next four rounds by a property of S-box. We extend such a 4-round characteristic by adding 3 rounds to the beginning and 2.5 rounds to the end of it. The best characteristic that we found for TWIS has probability 2^{-18} and is given in Table 1. For simplicity, we use the alternative round function given in [2]. In Table 1, the values ΔI_i refer to the input differences of the corresponding round. The output differences are not given additionally as they are the input differences of the next round.

Table 1. 9.5-round Differential Characteristic

Rounds	ΔI_0	ΔI_1	ΔI_2	ΔI_3	**# Active S-boxes**	**I/O Diff. for S-box**	**Probability**
1	02000000_x	00000000_x	00000000_x	$0000A600_x$	1	$0x02 \rightarrow 0xA6$	2^{-4}
2	00000000_x	00000000_x	01000000_x	00000000_x	1	$0x01 \rightarrow 0x00$	2^{-5}
3	01000000_x	00000000_x	00000000_x	00000000_x	1	$0x01 \rightarrow 0x00$	1*
4	00000000_x	00000000_x	00800000_x	00000000_x	0	-	1
5	00800000_x	00000000_x	00000000_x	00000000_x	0	-	1
6	00000000_x	00000000_x	00400000_x	00000000_x	0	-	1
7	00400000_x	00000000_x	00000000_x	00000000_x	0	-	1
8	00000000_x	00000000_x	00200000_x	00000000_x	1	$0x20 \rightarrow 0x83$	2^{-4}
9	00200000_x	00000000_x	80000041_x	00000000_x	2	$0x20 \rightarrow 0x83$ $0x01 \rightarrow 0x00$	2^{-5}*
9.5	80000041_x	80000041_x	00100000_x	00000000_x	1	$0x01 \rightarrow 0x00$	1*
	80000041_x	00004180_x	80100041_x	$C0000020_x$	-	-	

Notice that, in Table 1, the probability values of some rounds are marked with an asterisk(*) and these values are also relatively higher when considering the number of active S-boxes. The reason for high probability is that the cipher uses the same subkey for two consecutive G-functions and this makes them identical. To clarify, let x and $\bar{x}$ be two input values to G and y, $\bar{y}$ be the two corresponding output values. Then, if x and $\bar{x}$ are input to the next G-function which uses the same subkey, the outputs will again be y and $\bar{y}$. Hence, if an input pair with input difference Δx produces outputs with difference Δy with some probability p in G, then the same output difference Δy is produced with probability 1 when the input difference is Δx for the next G-function that uses the same subkey. Therefore, the probability of a differential characteristic that involves such G-functions is p instead of p^2. If each G-function were using different subkeys, the probability of the characteristic would be 2^{-32}.

2.2 10-round Differential Attack

We perform a key-recovery attack on 10-round TWIS, excluding the final key whitening, by using the 9.5-round differential characteristic given in Section 2.1 and recover 12 bits of the last round subkey RK_{10}. Adding a half round to the end of the given 9.5-round differential characteristic and simply tracing the differences, we obtain the difference between ciphertext pairs as (80100041_x,$C00041A0_x$, $????????_x$, 00418000_x).

The attack proceeds as follows:

1. Take $N = c.2^{18}$ plaintext pairs $P^i = (P_0^i, P_1^i, P_2^i, P_3^i)$, $P^{i*} = (P_0^{i*}, P_1^{i*}, P_2^{i*}, P_3^{i*})$ such that $P^i \oplus P^{i*} = (02000000_x, 00000000_x, 00000000_x, 0000A600_x)$ and obtain their corresponding ciphertexts $C^i = (C_0^i, C_1^i, C_2^i, C_3^i)$, $C^{i*} = (C_0^{i*}, C_1^{i*}, C_2^{i*}, C_3^{i*})$ by encrypting these plaintexts for 10 rounds of TWIS.
2. Check the first 64-bit and the last 32-bit ciphertext difference whether $C_0^i \oplus C_0^{i*} = 80100041_x$, $C_1^i \oplus C_1^{i*} = C00041A0_x$ and $C_3^i \oplus C_3^{i*} = 00418000_x$ and keep the text pairs satisfying these equations.

3. As the input differences of the S-boxes in the 10^{th} round are $0x3f \cdot 80 = 0x0$, $0x3f \cdot 10 = 0x10$, $0x3f \cdot 00 = 0x0$ and $0x3f \cdot 41 = 0x01$, one can attack the 2^{nd} and 4^{th} 8-bit words of RK_{10}. However, since two bits of each word vanish after bitwise AND operation, we can retrieve 12 bits of the subkey. Therefore, keep a counter for each possible value of the 12 bits of the subkey RK_{10} corresponding to the second and the fourth bytes.
4. Inputs of the last F-function are (C_0^i, RK_{10}) and $(C_0^{i^*}, RK_{10})$. XOR of output difference of this F-function and $((00418000_x) >>> 8)$ should be equal to the XOR of 80000041_x and $(\Delta C_2^i <<< 1)$. So, for each pair of plaintexts and their corresponding ciphertexts (C^i, C^{i^*}), increment the counter for the corresponding value of the subkey RK_{10} when the following equations holds:

$$F(C_0^i, RK_{10}) \oplus F(C_0^{i^*}, RK_{10}) \oplus 00004180_x = 80000041_x \oplus (\Delta C_2^i <<< 1).$$

5. Adopt the key with the highest counter as the right key.

The number of required plaintext pairs is $N = 4 \cdot 2^{18} = 2^{20}$ and this makes the data complexity of the attack 2^{21} chosen plaintexts. The time complexity of this attack is 2^{21} 10-round encryptions and the memory complexity is 2^{12}. Moreover, as the two attacked 6-bit words are independent from each other, one can keep two counters of 6 bits instead of a single counter of 12 bits, which reduces the memory complexity to 2^7.

The implementation of the attack verifies the results given in this section. Using the reference implementation of TWIS and taking $c = 4$, it takes only 15 seconds on a laptop[1] to get the 12 bits of the final subkey. By optimizing the reference code, the attack time can be decreased.

3 Impossible Differential Distinguisher for TWIS

While building the impossible differential characteristic, we were inspired from the differential characteristic given in Table 1. We combine two differential characteristics with probability one and obtain a contradiction by using the miss-in-the-middle approach[4]. The impossible differential characteristic is depicted in Figure 1, in which “0” denotes the 32-bit word consisting of all zeros.

In the left part of Figure 1, the input difference (0,0,Δy,0), Δy=00800000_x, is proceeded for 4.5 rounds in the forward direction and the difference (Δt,0,0,0), Δt=00200000_x, is obtained. On the other part, starting from the last round of the characteristic, the output difference (Δt,0,0,0) is traced backwards for 5 rounds and (0,0,Δx,0) difference where Δx=01000000_x, is acquired. However, we cannot have (Δt,0,0,0) = (0,0,Δx,0) since both Δt and Δx are non-zero differences. Therefore, (0,0,Δy,0) $\nrightarrow$ (Δt,0,0,0) after 9.5 rounds.

This characteristic can be extended to an impossible differential attack by adding half round to the beginning of the characteristic. By guessing the initial subkeys, wrong values can be eliminated and one will be left with the actual value of the subkeys.

4 Key Related Observations

This section is devoted to the observations on TWIS block cipher. These observations, which are mainly on key scheduling algorithm, include very basic design flaws like actual key size and trivial related key distinguishers that compromise the security of the algorithm.

[1] 2.2 Ghz Intel Core2Duo Processor, 2 GB Ram, Ubuntu 10.10 64 bit Operating System.

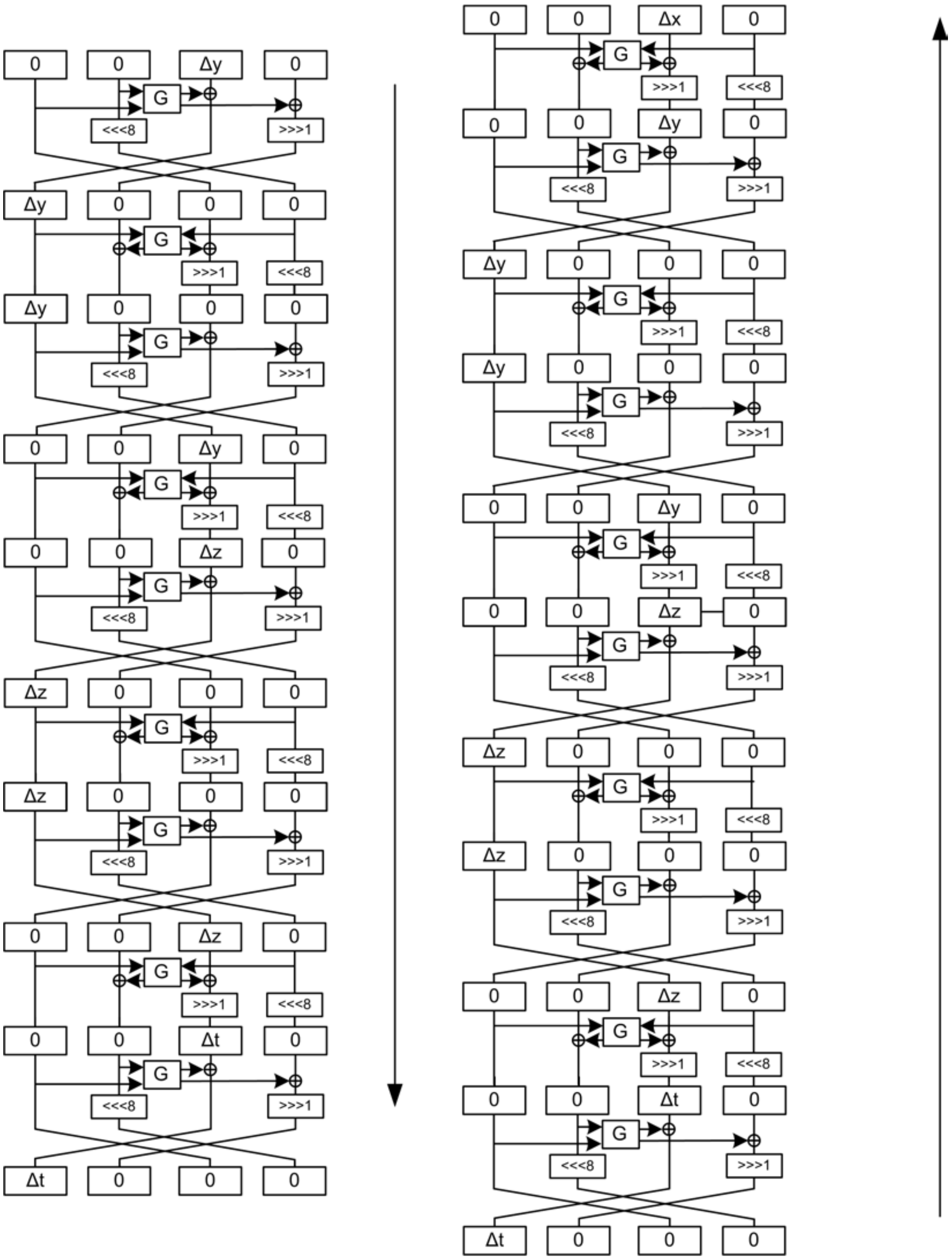


Fig. 1. Impossible Differential Characteristic where Δx=01000000$_x$, Δy=00800000$_x$, Δt=00200000$_x$, and Δz=00400000$_x$.

The most important flaw with the key schedule is that it does not use all bits of the master key. Instead, it uses only 54 bits of the 128-bit key. The first subkey is generated from the first 3 and the last 29 bits of the master key. Each remaining subkeys will be generated by left rotation of the modified key by 3. Therefore, key scheduling algorithm uses the first 33 and the last 29 bits of the key to derive the 11 subkeys which adds up 62 bits. Considering the bits eliminated by the S-Boxes in the key scheduling part, the actual key size of the cipher further reduces to 54 bits.

Another flaw arises from the S-box used in the key schedule. The S-box is used in the same manner with data processing part, so, one can find many related key distinguishers for TWIS. In order to form a related key distinguisher, it is enough to use a difference between two keys, where the difference coincides to the bit positions that are not processed by the S-box. The number of related key distinguishers can be increased by choosing the key differences that coincide the first two bit positions of 8-bit S-box input.

Also, in the data processing part, the data is XORed with the subkey and then S-box is applied to the XORed data. As S-box ignores the first two bits of the 8-bit input, 8 bits of the key is thrown away after this operation. So, the actual subkeys are 24 bits instead of 32 bits.

The key whitening is used in an inappropriate way. Notice that RK_0 is XORed to P_0 as the key whitening which also again XORed to P_0 in the first round inside the G-function. In this way RK_0 will be cancelled in G and it has no effect on the first G-function. Therefore, the cipher can be considered as 9,5 rounds.

Furthermore, the choice of final whitening subkeys results in a weakness. If one can determine the whole 32-bits of RK_2 and RK_{10} by attacking the final round, he can also determine the subkeys in between trivially

Besides, the diffusion of the key bits into the plaintext is not sufficient. This is a result of using an 8-bit word-wise permutation instead of a bitwise permutation and 8-bit S-box. This enables the attacker to mount an exhaustive search for a 32-bit subkey by dividing it into four 8-bit words without the knowledge of the remaining 24 bits. The complexity of such a search will be $4 \cdot 2^8 = 2^{10}$ instead of 2^{32}. However, in TWIS case, since the S-box ignores two input bits, one can recover the active subkey with $4 \cdot 2^6 = 2^8$ complexity.

5 Conclusion and Future Work

In this paper, we analyze the security of TWIS block cipher against differential, impossible differential attacks. Our results show that 10-round TWIS, when we exclude the final key whitening, is not resistant against differential attack. We recover half of the active key bits with 2^{21} chosen plaintexts. Also, we present distinguishers using the impossible differential technique. This distinguisher can be extended to key recovery attack. Finally, we propose some important observations on the algorithm.

As future a work, we aim to apply the mentioned attacks on full TWIS and mount related-key attacks by using the weaknesses in the key schedule.

References

1. Taizo Shirai, Kyoji Shibutani, Toru Akishita, Shiho Moriai, and Tetsu Iwata. The 128-Bit Blockcipher CLEFIA (Extended Abstract). In *FSE*, pages 181–195, 2007.
2. Bozhan Su, Wenling Wu, Lei Zhang, and Yanjun Li. Some Observations on TWIS Block Cipher. Cryptology ePrint Archive, Report 2010/066, 2010. http://eprint.iacr.org/.
3. Shrikant Ojha, Naveen Kumar, Kritika Jain, and Sangeeta Lal. TWIS - A Lightweight Block Cipher. In *ICISS*, pages 280–291, 2009.
4. Eli Biham, Alex Biryukov, and Adi Shamir. Miss in the Middle Attacks on IDEA and Khufu. In *FSE*, pages 124–138, 1999.

Breaking DVB-CSA

Erik Tews[1], Julian Wälde[1], and Michael Weiner[2]

[1] Technische Universität Darmstadt
{e_tews,jwaelde}@cdc.informatik.tu-darmstadt.de
[2] Technische Universtität München michaelweiner@mytum.de

Abstract. CSA (Common Scrambling Algorithm) is used to encrypt digital audio and video streams in DVB (Digital Video Broadcasting). This is commonly used to limit access to this content to paying customers. In this paper, we present a practical attack (time memory tradeoff) against CSA, that can be used to recover the ciphers key and decrypt the protected content. The attack is feasible against many currently deployed systems. We also discuss countermeasures that would defeat this attack without customer interaction.

1 Introduction CSA

CSA (Common Scrambling Algorithm) is the symmetric cipher used to protect content (mostly video data) in MPEG2 Transport Streams[3] in DVB (Digital Video Broadcasting). As part of the MPEG-TS standard, it is virtually used for all Conditional Access Systems in digital television.

1.1 Brief Description of the CSA Structure

A MPEG Transport Stream (TS) is a stream of 188 byte cells (4 byte header and 184 byte payload). Additional Header information can optionally be stored in the payload, using an Adaptation Field. In the following we will only look at cells that do not have additional header information. The payload can optionally be encrypted using one of two keys with DVB-CSA. One key is usually used to actually encrypt the payload while the other key is updated using a smart card or a similar system. Two "Scrambling Control" bits in the header indicate whether the cell was encrypted with the even(10) or the odd(11) key or whether it is unencrypted(00).

DVB-CSA encrypts in 2 steps. The first step splits the plain text in blocks of 64 bit length and encrypts them with a custom block cipher in CBC mode using reverse block order and an all zero initialization vector. In the second step, a custom stream cipher is used to encrypt all blocks except the first one, which is used as initialization vector. Note that CSA is completely deterministic and that this construction guarantees that every bit of the output depends on every bit of the input. Stream and block cipher share the same 64bit key.

[3] ISO 13818-1

1.2 Key Length

In spite of the fact that CSA works with 64 bit keys, we observed that only 48 bit of entropy are used for many TV stations. The fourth and the eigth byte of the key in this case are the sum of the previous three bytes:

$$\begin{array}{|c|c|c|c|c|c|c|c|}\hline k_0 & k_1 & k_2 & k_0+k_1+k_2 & k_4 & k_5 & k_6 & k_4+k_5+k_6 \\ \hline\end{array}$$

This reduces the effort needed for an exhaustive search from 2^{64} to 2^{48} trial decryptions.

This fact was not mentioned in previous academic publications [2–4] but is actually documented on the Wikipedia (as of 2006)[4]. Because CSA is a non public standard that has been reverse engineered, we do not know whether these checksums are part of the specification or originate from cryptography export restriction.

Since 2^{48} trial decryptions are clearly possible for small corporations and even individuals, CSA poses more likely a hindrance than a perfect protection of the payload. Most TV stations change the CSA key every 7 to 10 seconds using a smart card base key distribution system instead of using one (then manually entered) key over a longer period of time.

2 Rainbow Tables

Rainbow tables originally introduced by Oechslin [1] are a time-memory tradeoff that can be applied to find preimages for any given one-way function. A rainbow table used to compute preimages for $f(x)$ consists of the beginning and end of chains of the form:

$$k_0 \xrightarrow{R_0\circ f} k_1 \xrightarrow{R_1\circ f} k_2 \cdots k_{t-3} \xrightarrow{R_{t-3}\circ f} k_{t-2} \xrightarrow{R_{t-2}\circ f} k_{t-1}$$

R_i is a compression function family that takes a output of f and gives us an element of the domain in which we are looking for preimages of f.

$$i \neq j \Rightarrow R_i(x) \neq R_j(x)$$

Now given an output k the attacker tries every position in this chain from 0 to $t-1$ and computes the end of the chain from there until he finds a chain that has the same end as the one he just computed. The attacker now computes this chain from the beginning to the point of the preimage he is searching. Rainbow tables have lots of parameters that can be changed to meet restrictions imposed by the speed at which one can compute the one-way function $f(x)$ or the storage used.

[4] http://en.wikipedia.org/w/index.php?title=Common_Scrambling_Algorithm&diff=41583343&oldid=22087243

2.1 Known Plain text In MPEG2 Video Data

In H.262 (ISO 13818-2), the video compression codec used in DVB-S, so-called stuffing bytes are used to ensure a minimum bit rate. The ISO 13818-2 standard allows only zero bytes to be inserted between elements of the bit stream[5]. Since DVB-CSA is completely deterministic, encrypted all-zero MPEG-TS (Transport Stream) cells are detectable as collisions in the cipher text if two all-zero frames occur within the lifetime of a key. All-zero cells are mostly observed when a series of frames with no or only small differences is encoded. For example, a teleshopping show showing a static image of a product will generate lots of all-zero cells. A good counter-example would be a video recorded with an old film camera containing a lot of scratches or other artifacts. Some programs always contain a sufficient number of all-zero cells so that known plaintext can be recovered. We did not observe any other constant plaintext cells producing collisions in the transport stream.

As a one-way function upon which a rainbow table could be built, we propose a mapping f that takes an 48-bit key (without the two checksum bytes) as input and returns the first 6 ciphertext bytes of an all zero cell encrypted with this key. Note that in order to compute f, only 23 calls of the block cipher are required. As reduction function R_i one could simply xor the input of f with i.

3 High Speed Implementations of CSA

For the creation of the rainbow table, a fast implementations of the CSA block cipher would be helpful. We implemented the block cipher using the SSE2 instruction set of recent x86 CPUs. Also we have implementations using with CUDA and OpenCL that make use of the computational capacity of modern graphic accelerators.

3.1 Performance Data

A comparison of different implementations of CSA. By cell in this context we mean one evaluation of the one-way function.

Hardware	Implementation	cells/sec
GeForce GTX 460	OpenCL	3922848
GeForce GTX 460	CUDA	3391555
Intel Core i5 2.53 GHz	SSE	557103
AMD Phenom(tm) II X4 965 3.4 GHz	SSE	552486
AMD Phenom(tm) II X4 965 3.4 GHz	libdvbcsa	246913
Intel Core i5 2.53 GHz	libdvbcsa	217864

4 Rainbow Parameters

These high-speed implementations could be used to generate rainbow tables for various attack scenarios. Assuming that a hard-disk is able to perform 100 random accesses per second, and an adversary can encrypt about 4,000,000 cells on a GPU and about 500,000 cells on a single core CPU, we generated 3 parameter sets.

An adversary might be interested in recovering a DVB-CSA transmission in real-time. He needs to recover a single DVB-CSA key in less than 7 seconds. Using a GPU, the precomputations require 4 hard-disks and 7.9 TB of storage. Such a table can be precomputed on a single graphics card in less than 13 years. Using multiple graphics cards or faster graphics cards reduces the required time.

Alternatively, an adversary might not be interested in decoding a transmission in real time, or he would like to recover a static key from a station, that only changes the key manually. If a key should be recovered within 30 minutes, this can be done with 120GB of precomputations on a graphics card (less than 8 years of precomputations on a single graphics card) or 525GB of precomputations on a CPU (less than 5 years of precomputations on a graphics card).

Scenario	# Tables	# Chains	Chain-length	Coverage	Storage
GPU 7sec per key	2	$2^{38.488}$	2000	96.837%	7.9TB
GPU 30min per key	3	2^{32}	68410	91.953%	120GB
SSE 30min per key	18	$2^{31.542}$	10000	85.722%	525GB

5 Countermeasures

Even though the probably most secure solution to prevent this attack on CSA would be replacing it with a rather modern system; However, minor changes that would not required exchanging hardware in virtually every household in the western hemisphere, can also prevent this attack. For a start, using the 64 instead of 48 independent bits for a key would render time memory tradeoffs inefficient in comparison to the practice of Card Sharing [5]. Also, removing known plain text (sequences of 0-bytes longer that 183 bytes) in the video data would strike a devastating blow to the attack suggested here. The solution we think is best would be to send all zero filled cells selectively unencrypted while still using encryption for all other cells.

While the above is true for Conditional Access Systems (the biggest users of CSA) there is also the notion of more sensitive data being secured using CSA. With keeping in mind that 2^{64} trial decryptions required for exhaustive search of the key are still in reach of well funded organizations one might not want to rely on CSA if it comes to the transport of highly sensitive data via satellite.

6 Conclusion

This paper shows, that DVB-CSA can be broken in real time, using standard PC hardware, if precomputed tables are available. These precomputations can be performed on a standard PC in years. This makes DVB-CSA useless for any application, where real confidentiality is required. DVB-CSA might still be

[5] Somebody pays for access to the key material (usually a smart card of sorts) and then distributes the session keys via the Internet

used to protect digital content, where an adversary is not interested in attacks on the system, that recovers less than 99.9% of the payload. The attack can be prevented with small changes, what must be applied to the DVB-CSA encryption equipment, without having to alter the receivers side. We would like to thank everybody, who contributed to this paper.

References

1. Philippe Oechslin *Making a Faster Cryptanalytic Time-Memory Trade-Off* (2003)
2. Ralf-Philipp Weinmann Kai Wirt *Analysis of the DVB Common Scrambling Algorithm* (2004)
3. Kai Wirt *Fault Attack on the DVB Common Scrambling Algorithm* (2005)
4. Wei Li, Dawu Gu *Security Analysis of DVB Common Scrambling Algorithm* (2007)
5. ITU-T Video Coding Experts Group, ISO/IEC Moving Picture Experts Group *ISO/IEC 13818-2*

Critical attacks in code-based cryptography

Robert Niebuhr

Technische Universität Darmstadt
Fachbereich Informatik
Kryptographie und Computeralgebra,
Hochschulstraße 10
64289 Darmstadt
Germany
rniebuhr@cdc.informatik.tu-darmstadt.de

Abstract. In this paper we present a survey on critical attacks in code-based cryptography. In particular, we consider three cryptosystems – McEliece, Niederreiter, and HyMES – and analyze their vulnerability against a number of these attacks. All cryptosystems show a weakness against several attacks. We conclude with a discussion of techniques to protect against critical attacks.

Keywords: Code-based cryptography, critical attacks.

1 Introduction

In 1994, P. Shor [12] showed that quantum computers can break most "classical" cryptosystems, e.g. those based on the integer factorization problem or on the discrete logarithm problem. It is, therefore, crucial to develop cryptosystems that are resistant to quantum computer attacks. Cryptography based on error-correcting codes is a very promising candidate for post-quantum cryptography since code-based cryptographic schemes are usually fast and do not require special hardware, specifically no cryptographic co-processor.

Error-correcting codes have been applied in cryptography for at least three decades, ever since R. J. McEliece published his paper in 1978 [8]. The McEliece scheme is as old as RSA and has resisted cryptanalysis to date (except for a parameter adjustment). His work has received much attention as it is a promising candidate for post-quantum cryptography. Two other schemes that have recieved attention are the Niederreiter cryptosystem [11], and HyMES (Hybrid McEliece cryptosystem), developed by N. Sendrier and B. Biswas [3], which combines ideas from both schemes in order to increase the efficiency.

These cryptosystems are based on the syndrome decoding (SD) problem (or the general decoding problem, which can be reduced to it), which has been proved NP-complete [2]. There are generic attacks against these cryptosystems, e.g. based on information set decoding or the generalized birthday algorithm, buth these can be rendered infeasible by choosing appropriate parameters.

In practical applications, however, an attacker might not have to break the SD problem in order to decrypt a message. These critical attacks are possible usually when the attacker has some additional capability (e.g. a decryption oracle) or additional information (e.g. partial information about the plaintext).

Our contribution In this paper, we provide a survey of critical attacks against the three cryptosystems above and discuss techniques to protect against them.

Related work In [6], Kobara and Imai discuss some critical attacks (a subset of our list) against the McEliece cryptosystem and propose a conversion that protects against these attacks.

Organization of the paper Section 2 describes the three code-based cryptosystems. In Section 3 we present the different critical attacks and their application to the cryptosystems above. Section 4 concludes the paper.

2 Code-based encryption schemes

In this paper, a (n, k, t) code denotes a linear code of length n and dimension k capable of correcting t errors. If t is not relevant, we write (n, k) code for short. The (Hamming) weight of a vector v is denoted $\mathrm{wt}(v)$ and refers to the number of non-zero entries.

2.1 McEliece

The McEliece public-key encryption scheme was presented by R. McEliece in 1978 [8]. The original scheme uses binary Goppa codes, for which it remains unbroken (with suitable parameters), but the scheme can be used with any class of codes for which an effcient decoding algorithm is known.

Let G be a $k \times n$ generator matrix for a (n, k, t) Goppa code, P an $n \times n$ random permutation matrix, S a $k \times k$ invertible matrix and $\mathcal{D}_G$ a decoding algorithm for the code generated by G. All matrices and vectors are defined over a finite field $\mathbb{F}_q$.
The private key is $(S, G, P, \mathcal{D}_G)$, while $\widehat{G} = SGP$ and t are made public.

Encryption To encrypt a message $m \in \mathbb{F}_q^k$, the sender generates a random vector $e \in \mathbb{F}_q^n$ with $\mathrm{wt}(e) = t$ and computes the ciphertext $c = m\widehat{G} + e$.

Decryption Recieving a ciphertext c, the recipient computes $\widehat{c} = cP^{-1} = mSG + eP^{-1}$. Since P is a permutation, $\mathrm{wt}(eP^{-1}) = \mathrm{wt}(e)$, so $\mathcal{D}_G$ can be used to decode it: $mSG = \mathcal{D}_G(\widehat{c})$. The recipient then chooses a set $J \subseteq \{1, \dots, n\}$ such that $G_{\cdot J}$ (the matrix formed by the columns of G indexed by J) is invertible, and computes $m = mSG \cdot G_{\cdot J}^{-1} \cdot S^{-1}$.

2.2 Niederreiter

In 1986, H. Niederreiter proposed a cryptosystem [11] which can be seen as dual to the McEliece scheme. It uses the parity check matrix of a (usually Goppa) code to compute the syndrome of the message, which serves as the ciphertext. Even though the Niederreiter cryptosystem has been proven equally secure as the McEliece system [7], it is threatened by different critical attacks.

Let H be a $r \times n$ parity check matrix for a (n, k, t) Goppa code, where $r = n - k$, and $\mathcal{D}_H$ a decoding algorithm for the code defined by H. Since the underlying Goppa code can only correct a certain number $t < n$ of errors, the Niederreiter scheme uses a function φ to map the message to a word of length n and weight t: $\varphi : \mathbb{F}_q^l \mapsto \mathcal{W}_{n,t}$, where $l = \lceil \log_q \binom{n}{t} (q-1)^t \rceil$ and $\mathcal{W}_{n,t}$ is the set of vectors of length n and weight t.
The private key is $\mathcal{D}_H$, and H, t, and φ are made public.

Encryption Let $m \in \mathbb{F}_q^l$ be the message, then the ciphertext c is computed as $c = H \cdot \varphi(m)^T$.

Decryption Recieving a ciphertext c, the recipient decrypts it as $m = \varphi^{-1}(\mathcal{D}_H(c))$.

2.3 HyMES

The HyMES Hybrid McEliece cryptosystem developed by N. Sendrier and B. Biswas [3] increases the effciency of the McEliece scheme by encoding part of the message into the error vector. While in the usual scenario this scheme is as secure as the original McEliece scheme, it behaves differently facing critical attacks.

The HyMES scheme works as follows: The message m is split into two parts $m = (m_1|m_2)$. The first part m_1 corresponds to the message in the original McEliece scheme, while the second part is encoded into a word of weight t and serves as the error vector $e = \varphi(m_2)$.

Let G, P, S, and $\mathcal{D}_G$ be defined as for McEliece. Let φ be a function like in the Niederreiter scheme.
The private key is $(S, G, P, \mathcal{D}_G)$, while $\widehat{G} = SGP$, t, and φ are made public.

Encryption Let $m \in \mathbb{F}_q^{k+l}$ be the message, with l as above. Let m_1 be the first k bits of m and m_2 the remaining l bits. The ciphertext c is computed as $c = m_1\widehat{G} + \varphi(m_2)$.

Decryption The recipient first recovers m_1 as in the McEliece scheme above by computing $\widehat{c} = cP^{-1} = m_1SG + \varphi(m_2)P^{-1}$, applying the decoding algorithm $mSG = \mathcal{D}_G(\widehat{c})$, and finding $m_1 = mSG \cdot G_{\cdot J}^{-1} \cdot S^{-1}$ with J as above. The second part of m is found by computing $m_2 = \varphi^{-1}(c - m_1\widehat{G})$.

3 Critical attacks

3.1 Description

In this section, we give a brief description about the different critical attack we include in our analysis. The results are summarized in Figure 3.2.

Broadcast The general idea behind a broadcast scenario is that a sender send an identical message (or very similar messages) to a number of recipients. The message is encrypted with each recipient's own public key. A broadcast attack attempts to exploit the knowledge that the ciphertexts correspond to the same or similar messages in order to reveal the cleartext.
In 1988, J. Håstad [5] presented an attack against public key cryptosystems. This attack was originally aimed at the RSA cryptosystem, when a single message is sent to different recipients using their respective public keys. Håstad showed how to recover the message in this broadcast scenario. While this result is known for a long time, this type of attack has been considered only recently for cryptosystems based on error-correcting codes.
In [10], Niebuhr et al. presented a broadcast attack on the Niederreiter and HyMES cryptosystems that allowed to recover the cleartext in negligible time (10-20 seconds on a desktop PC) using only a small number of recipients – 3 for the Niederreiter parameters $(n, k) = (1024, 644)$.

Known partial plaintext This type of attack applies when an attacker knows part of the plaintext he attempts to reveal. This scenario arises in many applications, e.g. standardized emails or electronic forms.
The complexity of decoding the ciphertext decreases exponentially with every known bit. For example, attacking a ciphertext encrypted with McEliece using parameters (n, k) and knowing k_l bits is equivalent to attacking a McEliece ciphertext encrypted using parameters $(n, k - k_l)$. See [9,4,6] for more details.

Message-resend and related-message A message-resend condition is given if the same message is encrypted and sent twice (or several times) to the same recipient. If the subsequent messages are related to the first by a known relation, it is called a related-message condition. In the case of McEliece and HyMES, these conditions can be detected by observing the Hamming weight of the sum of two ciphertexts. By comparing the two ciphertexts, an attacker can identify those bits where

- with high probability, neither ciphertexts contains an error, or
- with certainty, exactly one contains an error.

This allows to recover the ciphertexts in negligible time [9].

(Adaptively) chosen ciphertext and Lunchtime In a chosen ciphertext attack, an attacker has access to a decryption oracle that allows to decrypt any chosen ciphertext (except the one the attacker attempts to reveal). In the general setting, the attacker has to choose all ciphertexts in advance before querying the oracle. In the adaptive chosen ciphertext attack, he is able to adapt this selection depending on the interaction with the oracle. A variant from the adaptive attack is the Lunchtime scenario which is derived from the idea that an attacker only has a limited time to access the oracle (e.g. while the victim is at lunch), e.g. the number of queries is limited.

Reaction attack This attack can be considered a weaker version of a chosen ciphertext attack. Instead of recieving the decrypted ciphertexts from the oracle, the attacker only observes the reaction of the oracle. Usually, this means whether the oracle was able to decrypt the ciphertext. In the context of side-channel-attacks, this can also mean observing decryption time, power consumption etc.
In one of the easiest variants, the attacker flips individual bits of the ciphertext he attempts to decode, and observes whether the oracle is able to decrypt it. If that is the case, the bit corresponds to an error bit (McEliece / HyMES). In the Niederreiter case, the same can be achieved by adding columns of the parity check matrix to the syndrome.

Malleability A cryptosystem is vulnerable to the malleability of it's ciphertexts if it is possible for an attacker to create new valid ciphertexts from a given one and if the new ciphertexts decrypts to a cleartext related to the original message. This property is relevant in many scenarios, e.g. bank transactions, where an attacker could change the amount of money transferred.

3.2 Results

The results of our analysis are summarized in Table 3.2.

4 Conclusion

In this paper we have analyzed the vulnerability of the McEliece, Niederreiter, and HyMES cryptosystems against several critical attacks. All schemes show a weakness against several of these attacks, HyMES against all of them. This result emphasizes the importance of conversions for these cryptosystems that protect against critical attacks. When choosing appropriate conversions, it is important to consider all critical attack above, since some conversions protect against only some of them; e.g. the well-known Optimal Asymmetric Encryption Padding (OAEP) by Bellare and Rogaway [1] is unsuitable for the McEliece/Niederreiter cryptosystems since it does not prevent reaction attacks. A secure conversion for the

		McEliece	Niederreiter	HyMES
Plaintext	Broadcast [10]	no	*	*
	Known partial [9]	*	*	*
	Message-resend [9]	*	no	*
	Related-message [9]	*	no	*
Ciphertext	Chosen	*	*	*
	Lunchtime	*	*	*
	Adapt. chosen [13]	*	*	*
	Reaction [6]	*	*	*
	Malleability [6]	*	no	*

Fig. 1. Critical attacks

McEliece cryptosystem has been proposed in [6], and for the Niederreiter cryptosystem in [6]. These are not applicable to the HyMES cryptosystem; however, the Mceliece conversion contains a similar technique as is used in HyMES, so the resulting scheme contains the efficiency improvements introduced in HyMES.

References

1. M. Bellare and P. Rogaway. Optimal asymmetric encryption. In *EUROCRYPT*, pages 92–111, 1994.
2. E. Berlekamp, R. McEliece, and H. van Tilborg. On the Inherent Intractability of Certain Coding Problems. *IEEE Transactions on Information Theory*, 24(3):384–386, 1978.
3. B. Biswas and N. Sendrier. Mceliece Cryptosystem Implementation: Theory and Practice. In *PQCrypto*, pages 47–62, 2008.
4. A. Canteaut and N. Sendrier. Cryptoanalysis of the original McEliece cryptosystem. In *ASIACRYPT*, pages 187–199, 1998.
5. J. Håstad. Solving simultaneous modular equations of low degree. *SIAM J. Comput.*, 17(2):336–341, 1988.
6. H. Imai and K. Kobara. Semantically Secure McEliece Public-Key Cryptosystems - Conversions for McEliece PKC. *Proc. of 4th International Workshop on Practice and Theory in Public Key Cryptosystems (PKC 01)*, pages 19–35, 2001.
7. Y.X. Li, R.H. Deng, and X.M. Wang. The equivalence of McEliece's and niederreiter's public-key cryptosystems. *IEEE Trans. Inform. Theory*, 40:271–273, 1994.
8. R. McEliece. A Public-Key Cryptosystem Based on Algebraic Coding Theory. The Deep Space Network Progress Report, DSN PR 42–44, 1978. `http://ipnpr.jpl.nasa.gov/progressreport2/42-44/44N.PDF`.
9. R. Niebuhr. *Application of Algebraic-Geometric Codes in Cryptography.* Vdm Verlag Dr. Müller, 2008.
10. R. Niebuhr and P.-L. Cayrel. Broadcast attacks against code-based encryption schemes. Submitted to IWSEC, 2011.
11. H. Niederreiter. Knapsack-type Cryptosystems and Algebraic Coding Theory. *Problems of Control and Information Theory*, 15(2):159–166, 1986.
12. P. W. Shor. Polynomial-Time Algorithms for Prime Factorization and Discrete Logarithms on a Quantum Computer. *SIAM Journal on Computing.*
13. H.-M. Sun. Further cryptanalysis of the mceliece public-key cryptosystem. *IEEE Communications Letters*, 4(1):18–19, Jan 2000.

A taxonomy of non-cooperatively computable functions
(Extended Abstract)

Yona Raekow and Konstantin Ziegler

B-IT, Universität Bonn
D-53113 Bonn, Germany
{raekow, zieglerk}@bit.uni-bonn.de

Abstract. Multi-party computation deals with enabling players to jointly compute a function over their private inputs. In this work we consider players which behave rationally, i.e. all players want to maximize their own profit. The question is, if there exist functions where players achieve a Nash-equilibrium if they submit their true input and such enable all players to compute the correct value of the function. Such functions are called non-cooperatively computable. In this paper, we analyze whether non-cooperatively computable Boolean functions exist and we define properties of such functions. In order to define the pay-off of the players we state the preferences that players which participate in a cryptographic multi-party computation protocol have. We show how many Boolean functions are non-cooperatively computable if players prefer to compute the correct outcome and secondly prefer that the other players cannot compute the correct outcome. Furthermore, we analyze functions that are privacy-preserving, i.e. that do not reveal anything about the private inputs of the players. We conjecture that there are no Boolean functions that are privacy-preserving and that are non-cooperatively computable.

1 Introduction

Introduced in [4], non-cooperative computation (NCC) deals with the joint evaluation of a multivariate function by rational (i.e. self-interested) players, where each player provides one of the input values. Players communicate their input to a trusted center, which performs a commonly known computation and returns the result to the players. We analyze functions that admit NCC. The question is whether the players can be incented to communicate their inputs correctly to the center and believe that the value returned by the center is the output of the function on correct inputs.

In order to answer this question one has to identify preferences that players might have and their respective orderings. In [2, 3] the following preferences for cryptography are listed:

correctness: the player wants to compute the correct outcome.
exclusivity: as few other players as possible learn the correct outcome.
privacy: the player does not want other players to learn anything about his input.
voyeurism: the player learns as much as possible about the inputs of the other players.

The authors in [4] provide a classification of Boolean functions that are non-cooperative computable provided that the players value correctness first and exclusivity second. They give concrete examples of Boolean functions that are not non-cooperatively computable, such as the XOR function.

In this paper we look at Boolean functions that actually *are* non-cooperatively computable, provide examples and a thorough analysis for small numbers of players.

The game-theoretic approach has also proven to be useful in the setting of multi-party computation and secret sharing. Typically protocols in these fields are designed under the assumption that there are good and bad players, where the good ones follow the protocol and the bad ones do not. Assuming players that instead value correctness over exclusivity, [1] show that multi-party computations without trusted center and secret sharing is impossible with protocols of fixed running time.

2 Boolean players and functions

Let $n \geq 1$ be the number of players. Every player P_i has a *type* T_i drawn from the set $B = \{0, 1\}$ according to a probability distribution Δ_i. We simply write t_i for $T_i = t_i$ and assume full support for Δ_i, i.e. $\Delta_i(t_i) > 0$ for all t_i. Combining the types of all players yields the vector $t \in B^n$ and omitting the type of player P_i the vector $t_{-i} \in B^{n-1}$. The set of Boolean functions in n variables is denoted by $\mathcal{F}_n = \{F \colon B^n \to B\}$. It has size $\#\mathcal{F}_n = 2^{(2^n)}$. For any Boolean function F, we have its *complement* $\overline{F}$ described by $\overline{F}(t) = \overline{F(t)} = F(t) \oplus 1$.

We will list several properties of interest and follow up with examples and a discussion of the respective numbers for small n.

P_0	P_1	F_0	F_1	F_2	F_3	F_4	F_5	F_6	F_7	F_8	F_9	F_{10}	F_{11}	F_{12}	F_{13}	F_{14}	F_{15}
0	0	0	0	0	0	0	0	0	0	1	1	1	1	1	1	1	1
0	1	0	0	0	0	1	1	1	1	0	0	0	0	1	1	1	1
1	0	0	0	1	1	0	0	1	1	0	0	1	1	0	0	1	1
1	1	0	1	0	1	0	1	0	1	0	1	0	1	0	1	0	1

Table 1. All Boolean functions $F \colon B^2 \to B$.

Definition 1 (degenerate). *Let $F \in \mathcal{F}_n$. The player P_i is called* relevant, *if there exist $t_i \in B$ and $t_{-i} \in B^{n-1}$, such that*

$$F(t_i, t_{-i}) \neq F(\overline{t_i}, t_{-i}).$$

Functions, where some players are not relevant are called degenerate.

Clearly, the function $\texttt{constant}_t$, that returns the constant value t for any input vector, is always degenerate. In two variables (Table 1), these are F_0 and F_{15}.

Definition 2 (conditionally/absolutely dominated). *Let $F \in \mathcal{F}_n$. A player P_i is called* conditionally *(or* absolutely*)* dominating, *if*

1. *P_i is relevant and*
2. *there is some $t_i \in B$, such that for all $y_{-i}, z_{-i} \in B^{n-1}$, we have*

$$F(t_i, y_{-i}) = F(t_i, z_{-i})$$

(and also $F(\overline{t_i}, y_{-i}) = F(\overline{t_i}, z_{-i})$).

If such a player exists, we call F conditionally *(or* absolutely*)* dominated.

The only absolutely dominated functions are $\texttt{dictator}_i$, which simply return the input of player P_i, and their complements. In two variables (Table 1), these are F_3 and F_5, respectively.

Definition 3 (reversible). *Let $F \in \mathcal{F}_n$. We say, that a player P_i* can reverse *F, if for every $t_{-i} \in B^{n-1}$*

$$F(t_i, t_{-i}) = \overline{F(\overline{t_i}, t_{-i})}.$$

If such a player exists, we call F reversible.

Clearly, a player who is able to reverse F has to be relevant. The authors of [4] note that the only *symmetric* reversible functions are $\texttt{parity}$, returning the XOR of the inputs, and its complement. In two variables (Table 1), the parity function is F_6 and its complement is F_9. Furthermore, a function is reversible and conditionally dominated if and only if it is $\texttt{dictator}_i$ for some i or the complement.

Definition 4 (privacy-preserving). *Let $F \in \mathcal{F}_n$. We say that player P_j* can violate the privacy *of player P_i with $i \neq j$, if there is an input $t_j \in B$ and values x,$y \in B$, such that*

$$F(t_j, t_{-j}) = x \Rightarrow (t_i = y)$$

for all $t_{-j} \in B^{n-1}$. If such a pair of players exists, we say that F admits privacy violations *and otherwise that F is* privacy-preserving*.*

Figure 1 puts properties in relation to each other.

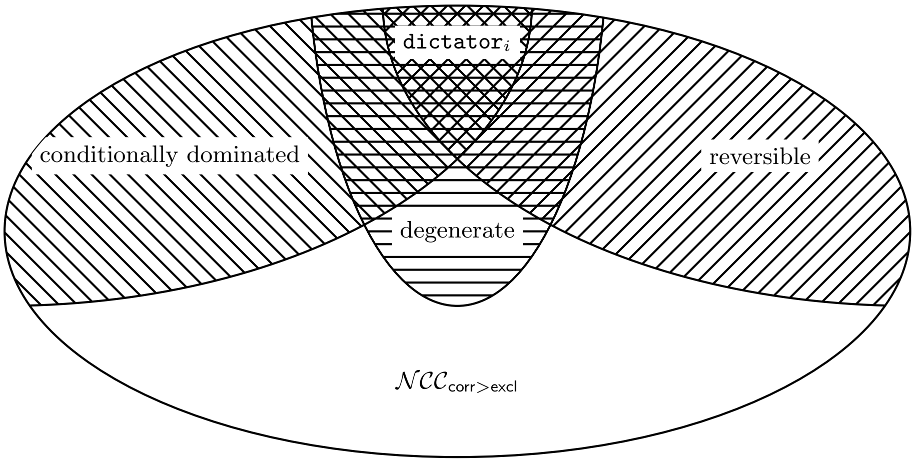


Fig. 1. Properties of Boolean functions.

We remark the following criterion.

Proposition 1. *A Boolean function is degenerate/conditionally dominated/reversible/privacy preserving, if and only if its complement is.*

We conclude with Table 2, listing the number of Boolean functions satisfying the named criteria. The number of degenerate Boolean functions is sequence A005530 in the OEIS [5].

n	$\#\mathcal{F}(n)$	degenerate	non-degenerate	conditionally dominated	reversible
1	4	2	2	2	2
2	$2^{(2^2)} = 16$	6	10	8	2
3	$2^{(2^3)} = 256$	38	218	118	2
4	$2^{(2^4)} = 65536$	942	64594	3512	934

Table 2. The numbers of functions that are degenerate, non-degenerate, conditionally dominated and reversible for small n.

3 Non-cooperative computations

We consider the non-cooperative computation of a Boolean function for at least 2 rational players. We assume that all players are relevant to the function, or in other words, that the function is non-degenerate.

Each player participating in a computation has to make two strategic choices.

- Which value $g_i(t_i)$ should he report to the center?
- Which value $f_i(F(g_j(t_j)), t_i)$ should be his guess for $F(t)$)?

A simple strategy is to report truthfully, $g_i(t_i) = t_i$, and to believe the center's return value, $f_i(F(g_j(t_j)), t_i) = F(g_j(t_j))$. This strategy is called *straight-forward* and a function F is called *non-cooperatively computable* if the game is in a Nash-equilibrium when everybody plays the straight-forward strategy. In other words, no player can achieve a more favorable outcome by deviating from the straight-forward strategy, assuming that all other players use it. We discuss two particularly interesting combinations of "favorable outcomes".

3.1 Correctness first, exclusivity second

The following theorem is due to Shoham and Tennenholtz [4]:

Theorem 1 **($\mathcal{NCC}_{\mathsf{corr}>\mathsf{excl}}$).** *If players value correctness first and exclusivity second then a Boolean function is non-cooperative computable if it is not reversible and not conditionally dominated.*

We emphasize the importance of excluding degenerate functions. Otherwise, the constant functions would also satisfy the criterion above. We used this theorem to determine whether such functions exist. We answer this in the affirmative for $n = 3$ and 4 (Table 3). Note, however we found that there are no $\mathcal{NCC}_{\mathsf{corr}>\mathsf{excl}}$-functions for two players.

3.2 Privacy-preserving functions

In this section we take the work in [4] one step further: While the authors in [4] considered only the preferences correctness over exclusivity, we consider in this section the additional preference privacy. At first, we consider the privacy of one particular P_i and we denote the set of functions that preserves its as

$$\mathcal{P}riv_i = \{F : \text{no } P_j, j \neq i, \text{ can violate the privacy of } P_i\}.$$

Next, we look at the intersection of functions that do not admit any privacy violations, we denote it as $\mathcal{P}riv$ i.e.

$$\mathcal{P}riv = \bigcap_{i=0}^{n} \mathcal{P}riv_i.$$

Taking a closer look at the functions in $\mathcal{P}riv_i$, it is interesting to see that this set contains exactly the functions that are degenerate for player P_i, as well as the parity function and its complement. Therefore, the only functions in the set $\mathcal{P}riv$ are the parity function, its complement, and the constant functions. Note, that constant functions are degenerate for any player.

Table 3 shows how many functions do not admit privacy violations for one particular player ($\mathcal{P}riv_i$) or for any player ($\mathcal{P}riv$), respectively for $n = 2, 3, 4$. Based on our experimental results we state the following conjecture: There are no privacy preserving functions that are also $\mathcal{NCC}_{\mathsf{corr}>\mathsf{excl}}$ in the sense of [4].

Theorem 6 of [2] states that "If privacy is ranked over correctness, and both are ranked over exclusivity, then a function is NCC if and only if it is not reversible, non-dominated and has no privacy violations." As shown above, no such functions exist for $n \leq 4$, since the only functions that do not admit privacy violations are degenerate and therefore dominated.

4 Outlook

To complete the picture of non-cooperatively computable Boolean functions, we need to carefully define and analyze the functions allowing voyeurism. We observed that the four preferences correctness, exclusivity, privacy and voyeurism suggested for cryptography are fairly generic and that further refinements may be in place. Non-cooperatively computable Boolean functions are an

n	total	$\mathcal{NCC}_{\mathsf{corr}>\mathsf{excl}}$
2	10	0
3	218	106
4	64594	61090

n	total	$\mathcal{P}riv_i$	$\mathcal{P}riv$
2	16	4	2
3	256	18	4
4	65536	18976	4

Table 3. The number of $\mathcal{NCC}_{\mathsf{corr}>\mathsf{excl}}$ functions in relation to all non-degenerate Boolean functions. The number of functions that do not admit privacy violations for a particular player P_i or any player, respectively in relation to all Boolean functions.

important first step to identify suitable functions for multi-party computation. If rational players do not want to comply with the protocols, since the benefit for them is higher if they do not follow the rules, then even the best cryptographic protocol fur multi-party computation is useless. Boolean functions are, however, only a small set of functions. It would be interesting to investigate functions with larger domains and ranges. Furthermore, we assume that the input types of the players are distributed independently, an interesting case would also be to investigate correlated inputs. The authors in [2] investigated all orderings among the four preferences. We believe it is important to investigate all subsets of the four preferences and also to consider that players participating in the same game might have different preferences.

5 Acknowledgments

The work of Yona Raekow and Konstantin Ziegler was supported by the B-IT Foundation and the Land Nordrhein-Westfalen.

References

1. Halpern, J.Y., Teague, V.: Rational secret sharing and multiparty computation: extended abstract. In: STOC. pp. 623–632 (2004)
2. McGrew, R., Porter, R., Shoham, Y.: Towards a general theory of non-cooperative computation (extended abstract). In: TARK. pp. 59–71 (2003)
3. Nisan, N., Roughgarden, T., Tardos, E., Vazirani, V.V. (eds.): Algorithmic Game Theory. Cambridge University Press, New York, NY, USA (September 2007)
4. Shoham, Y., Tennenholtz, M.: Non-cooperative computation: Boolean functions with correctness and exclusivity. Theor. Comput. Sci. 343(1-2), 97–113 (2005)
5. The OEIS Foundation: A005530 Number of Boolean functions of n variables from Post class F(8,inf); number of degenerate Boolean functions of n variables. http://oeis.org/A005530 (Jun 2011), [Online; accessed 04-Jun-2011]

The preimage security of double-block-length compression functions

Frederik Armknecht[1], Ewan Fleischmann[2], Matthias Krause[1], Jooyoung Lee[3], Martijn Stam[4], and John Steinberger[5]*

[1] Arbeitsgruppe Theoretische Informatik und Datensicherheit, University of Mannheim, Germany,{armknecht,krause}@uni-mannheim.de
[2] Chair of Media Security, Bauhaus-University Weimar, Germanyewan.fleischmann@uni-weimar.de
[3] Faculty of Mathematics and Statistics, Sejong University, Seoul, Korea, jlee05@sejong.ac.kr
[4] Dept. of Computer Science, University of Bristol, United Kingdom, m.stam@alumnus.tue.nl
[5] Institute of Theoretical Computer Science, Tsinghua University, Beijing, China, jpsteinb@gmail.com

Abstract. We present new techniques for deriving preimage resistance bounds for block cipher based double-block-length, double-call hash functions. We give improved bounds on the preimage security of the three "classical" double-block-length, double-call, block cipher-based compression functions, these being Abreast-DM, Tandem-DM and Hirose's scheme. For Hirose's scheme, we show that an adversary must make at least 2^{2n-5} block cipher queries to achieve chance 0.5 of inverting a randomly chosen point in the range. For Abreast-DM and Tandem-DM we show that at least 2^{2n-10} queries are necessary. These bounds improve upon the previous best bounds of $\Omega(2^n)$ queries, and are optimal up to a constant factor since the compression functions in question have range of size 2^{2n}.

Keywords: Hash Function, Preimage Resistance, Block Cipher, Beyond Birthday Bound, Foundations

1 Introduction

Almost as soon as the idea of turning a block cipher into a hash function appeared [14], it became evident that, for typical block ciphers and security expectations, the hash function needs to output a digest that is considerably larger than the block cipher's block size. Consequently, many proposals of double-block-length, or more generally multi-block-length, hash functions have appeared in the literature. In this article we focus on a subclass of double-block-length constructions, where a $3n$-bit to $2n$-bit compression function makes two calls to a block cipher of $2n$-bit key and n-bit block.

Recently, for all three well-known members of this class—those being Tandem-DM [8], Abreast-DM [8] and Hirose's construction [6]—collision resistance has been successfully resolved [4, 6, 9, 10]: for Abreast-DM and Hirose's scheme, $\Omega(2^n)$ queries to the underlying block cipher are needed to obtain a non-vanishing advantage in finding a collision. For Tandem-DM, $\Omega(2^{n-\log n})$ queries are needed, which is almost optimal ignoring log factors.

On the other hand, the corresponding situation for preimage resistance is far less satisfactory. Up to now, it has been an open problem to prove preimage resistance for values of q higher than 2^n for either Abreast-DM, Tandem-DM or Hirose. This is not to say that no dedicated preimage security proofs have appeared in the literature. For instance, Lee, Stam and Steinberger [10] provide a preimage resistance bound for Tandem-DM that is a lot closer to 2^n than a straightforward implication [15] of their collision bound would give. However, a "natural barrier" occurs once 2^n queries are reached: namely, a block cipher "loses randomness" after being queried $\Omega(2^n)$ times on the same key (for example, when $2^n - 1$ queries have been made to a block cipher under a given key, the answer to the last query under that key is deterministic). Going beyond the 2^n barrier seemed to require either a very technical probabilistic analysis, or some brand new idea. In this paper, we show a new idea which delivers tight bounds in a quite pain-free and non-technical fashion.

* Supported by the National Natural Science Foundation of China Grant 61033001, 61061130540, 61073174, by the National Basic Research Program of China Grant 2007CB807900, 2007CB807901 and by NSF grand CNS 0904380.

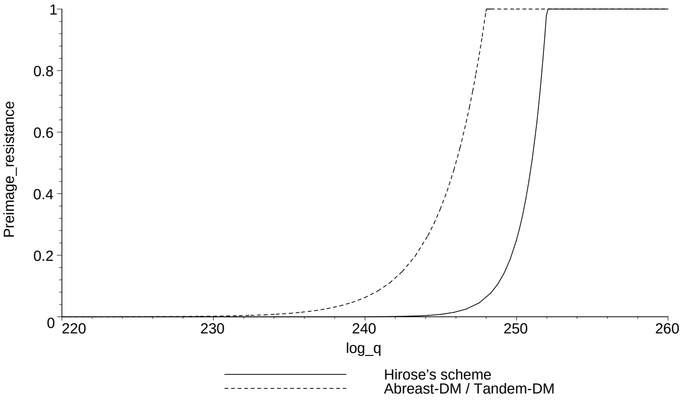


Fig. 1: Preimage bounds for the classical constructions.

OUR CONTRIBUTION. In this paper, we prove that various compression functions that turn a block cipher of $2n$-bit key into a double-block-length hash function, have preimage resistance close to the optimal 2^{2n} in the ideal cipher model. Our analysis covers many practically relevant proposals, such as Abreast-DM, Hirose-DM and Tandem-DM. Bounds for the case $n = 128$ are depicted in Figure 1. At the heart of our result are so-called "super queries", a new technique to restrict the advantage of an adaptive preimage-finding adversary.

To build some intuition for our result, let us start with considering the much easier problem of constructing a $3n$-bit to $2n$-bit compression function H based on two $3n$-bit to n-bit smaller underlying primitives f and f'. An obvious approach is simply to concatenate the outputs of f and f', that is let $H(B) = f(B)\|f'(B)$ for $B \in \{0,1\}^{3n}$. If f and f' are modeled as independently sampled, ideally random functions, then it is not hard to see that H behaves ideally as well. In particular, it is preimage resistant up to 2^{2n} queries (to f and f').

When switching to a block cipher-based scenario, it is natural to replace f and f' in the construction above by E, resp. E', both run in Davies–Meyer mode. In other words, for block ciphers E and E' both with $2n$-bit keys and operating on n-bit blocks, define $H(A\|B) = (E_B(A) \oplus A)\|(E'_B(A) \oplus A)$ where $A \in \{0,1\}^n$ and $B \in \{0,1\}^{2n}$. While there is every reason to believe this construction maintains preimage resistance up to 2^{2n} queries, the standard proof technique against adaptive adversaries falls short significantly. Indeed, the usual argument goes that the i-th query an adversary makes to E using key K will return an answer uniform from a set of size at least $2^n - (i-1)$ and thus the probability of hitting a prespecified value is at most $1/(2^n - (i-1)) < 1/(2^n - q)$. Unfortunately, once q approaches 2^n, the denominator tends to zero (rendering the bound useless). As a result, one cannot hope to prove anything beyond 2^n queries using this method. This restriction holds even for a "typical" bound of type $q/(2^n - q)^2$.

When considering *non-adaptive* adversaries only, the situation is far less grim. Such adversaries need to commit to all queries in advance, which allows bounding the probability of each individual query hitting a prespecified value by 2^{-n}. While obviously there are dependencies (in the answers), these can safely be ignored when a union bound is later used to combine the various individual queries. Since the q offset has disappeared from the denominator, the typical bound $q/(2^n)^2$ *would* give the desired security.

Our solution, then, is to force an adaptive adversary to behave non-adaptively. As this might sound a bit cryptic, let us be more precise. Consider an adversary adaptively making queries to the block cipher, using the same key throughout. As soon as the number of queries *to this key* passes a certain threshold, we give the remaining queries to the block cipher using this very key *for free*. We will refer to this event as a *super query*. Since these free queries are all asked in one go, they can be dealt with non-adaptively, preempting the problems that occur (in standard proofs) due to adaptive queries. Nonetheless, for every super query we need to hand out a very large number of free queries, which can aid the adversary. Thus we need to limit the amount of super queries an adversary can make by setting the threshold that triggers a super query sufficiently high. In fact, we set the threshold at exactly half[6] the total number of queries that can be made under a given key (i.e., it is set at $2^n/2$ queries). This effectively doubles the adversary's query budget, since for every query the adversary makes it can get another one later "for free" (if it keeps on making queries under the same key), but such a doubling of the number of queries does not lead to an unacceptable deterioration of the security bound.

With this new technique in hand, we prove that the construction H given above has indeed an asymptotically optimal preimage resistance bound (a generalization of this result is also given) We revisit the proofs of preimage resistance of the three main double-block-length, double-call constructions: Hirose, Abreast-DM and Tandem-DM. An additional technical problem is that these compression functions each make two calls to the same block cipher, as opposed to using two calls to independent block ciphers. Ideally, to get a good bound, one would like to query the two calls necessary for a single compression function evaluation in conjunction (this would allow using the randomness of both calls simultaneously, potentially leading to a denominator 2^{2n} as desired for preimage resistance). For instance, in the context of collision resistance for Hirose-DM and Abreast-DM corresponding queries are grouped in cycles (of length 2 and 6, respectively) and all queries in a cycle are made simultaneously: if the adversary makes one query in a cycle, the remaining queries are handed out for free. Care has to be taken that these free queries and the free queries due to super queries do not reinforce each other to untenable levels.

For Hirose's scheme, there are no problems as the free queries introduced by a super query necessarily consist of full cycles only. The corresponding (upper) bound on the preimage finding advantage is $16q/2^{2n}$ which is as desired, up to a small factor. For Abreast-DM, however, the cyclic nature can no longer be exploited: any super query introduces many partial cycles, yet freely completing these might well trigger a new super query, etc.! Luckily, the original preimage proof for Tandem-DM [10] (which does not involve cycles) provides a way out of this conundrum. The downside however is that our preimage bound for Abreast-DM and Tandem-DM is slightly less tight than that for Hirose's scheme. Ignoring negligible terms, it grows roughly as $16\sqrt{q}/2^n$. Although this is faster than one might wish for, it does imply that $\Omega(2^{2n})$ queries are required to find a preimage with constant probability.

References

1. Y. Dodis and J. Steinberger: Message Authentication Codes from Unpredictable Block Ciphers. Crypto 2009, LNCS 5677, pp. 267–285. Springer, Heidelberg (2010). Full version available at `http://people.csail.mit.edu/dodis/ps/tight-mac.ps`
2. E. Fleischmann, C. Forler, M. Gorski and S. Lucks: Collision Resistant Double-Length Hashing. ProvSec 2010, LNCS 6401, pp. 102–118. Springer, Heidelberg (2010)
3. E. Fleischmann, M. Gorski and S. Lucks: On the security of Tandem-DM. FSE 2009, LNCS 5665, pp. 84–103. Springer, Heidelberg (2009)
4. E. Fleischmann, M. Gorski and S. Lucks: Security of cyclic double block length hash functions. Cryptography and Coding, 12th IMA International Conference, Cirencester, UK, LNCS 5921 pp. 153–175. Springer, Heidelberg (2009)
5. S. Hirose: Provably secure double-block-length hash functions in a black-box model. ICISC 2004, LNCS 3506, pp. 330–342. Springer, Heidelberg (2005)

[6] The "optimized" threshold turns out to be very near one half, but a bit less; we set the threshold at a half for simplicity in our proofs.

6. S. Hirose: Some plausible constructions of double-block-length hash functions. FSE 2006, LNCS 4047, pp. 210–225. Springer, Heidelberg (2006)
7. M. Krause, F. Armknecht and E. Fleischmann: Preimage resistance beyond the birthday bound: Double-length hashing revisited. `http://eprint.iacr.org/2010/519.pdf`
8. X. Lai and J. Massey: Hash function based on block ciphers. Eurocrypt 1992, LNCS 658, pp. 55–70. Springer, Heidelberg (1993)
9. J. Lee and D. Kwon: The security of Abreast-DM in the ideal cipher model. `http://eprint.iacr.org/2009/225.pdf`
10. J. Lee, M. Stam and J. Steinberger: The security of Tandem-DM in the ideal cipher model. `http://eprint.iacr.org/2010/409.pdf`
11. J. Lee and J. Steinberger: Multi-property preservation using polynomial-based modes of operation. Eurocrypt 2010, LNCS 6110, pp. 573–596. Springer, Heidelberg (2010)
12. S. Lucks: A collision-resistant rate-1 double-block-length hash function. Symmetric Cryptography, Dagstuhl Seminar Proceedings 07021 (2007)
13. O. Özen and M. Stam: Another Glance at Double-Length Hashing. Cryptography and Coding, 12th IMA International Conference, Cirencester, UK, LNCS 5921, pp. 94–115. Springer, Heidelberg (2009)
14. M. Rabin: Digitalized signatures. Foundations of Secure Computations, pages 155–166. Academic Press (1978)
15. P. Rogaway and T. Shrimpton: Cryptographic hash-function basics: definitions, implications, and separations for preimage resistance, second-preimage resistance, and collision-resistance. FSE 2004, LNCS 3017, pp. 371–388. Springer, Heidelberg (2004)
16. P. Rogaway and J. Steinberger: Constructing cryptographic hash functions from fixed-key block ciphers. Crypto 2008, LNCS 5157, pp. 433–450. Springer, Heidelberg (2008)
17. T. Shrimpton and M. Stam: Building a collision-resistant compression function from non-compressing primitives. ICALP 2008, Part II. LNCS 5126, pp. 643–654, Springer, 2008.
18. J. Steinberger: The collision intractability of MDC-2 in the ideal-cipher model, Eurocrypt 2007, LNCS 4515, pp. 34–51. Springer, Heidelberg (2007)
19. M. Stam: Beyond uniformity: better security/efficiency tradeoffs for compression functions, Crypto 2008, LNCS 5157, pp. 397–412. Springer, Heidelberg (2008)
20. M. Stam: block cipher-based hashing revisited, FSE 2009, LNCS 5665, pp. 67–83. Springer, Heidelberg (2009)
21. D. Wagner: Cryptanalysis of the Yi-Lam hash, Asiacrypt 2000, LNCS 1976, pp. 483–488. Springer, Heidelberg (2000)
22. X. Yi and K.-Y. Lam: A new hash function based on block cipher. ACISP 1997, Second Australasian Conference on Information Security and Privacy, LNCS 1270, pp. 139–146. Springer, Heidelberg (1997)

Algorithmically determining the optimum polynomial multiplier in $GF(2^k)$

Zoya Dyka and Peter Langendoerfer

IHP,
Im Technologiepark 25, D-15232 Frankfurt (Oder), Germany
http://www.ihp-microelectronics.com/

Abstract. In this paper we present an algorithm that determines the segmentation of operands for polynomial multiplication and computes the optimal combination of multiplication methods. The result of the algorithm can be used to implement highly efficient hardware multipliers in $GF(2^k)$. Our calculation shows that thise implementations are about 10 per cent smaller than the most efficient multipliers known from literature.

Keywords: ECC, polynomial multiplication, hardware implementation

1 Introduction

During recent years elliptic curve cryptography (ECC) has gained significant attention especially for devices such as wireless sensor nodes. Due to their scarce resources hardware implementations are considered important. The polynomial multiplication is the operation which is investigated most since it is one of the most complex field operations and executed very often.

There exist many multiplication methods (MMs) for polynomiasl over $GF(2^k)$ that apply segmentation of both k-bit long multiplicands into n parts (terms): the classical and the generalized Karatsuba MM for $n > 1$; Karatsuba MM for 2- and Winograd MM for 3-term operands, that are both the special cases of the generalized Karatsuba MM; Montgomery MM for 5-, 6- and 7- term operands and many other MMs. These MMs lead to a reduced number of partial multiplications but require more XOR-operations in comparison to the classical MM.

The reduction of the number of partial multiplications improves notin all cases the chip-parameter (area, energy consumption) of the multiplier. For small operands, the classical MM is the favorite. A combination of classical MM for calculation of small partial products with other MM can improve chip-parameters of the resulting multipliers [1]. An additional means to improve the chip-parameters of the multipliers is the reduction of the number of additions (XOR-operations). The reduction can be achieved by using pre-defined processing sequences for additions of partial products [2]. If an optimal combination of several multiplication approaches with the reduced number of XOR-operations is found, the area and energy consumption is reduced significantly.

This paper presents an algorithm that determines the optimal combination of multiplication methods for which an optimized processing sequence is already pre-defined. For the assessment of the chip parameters we use XOR and AND gates. We are aware of the fact that gate properties are technology dependent. By initializing our algorithm with the specific area or energy consumption of used gates it can be applied for each technology.

The rest of paper is structured as follows: in section 2 we give the exact complexity of the six investigated MMs for their original processing sequences and for our pre-defined optimized processing sequences. In addition we introduce our algorithm to determine their optimal combination. The evaluation of our results is discussed in section 3. The paper concludes with a short summary.

2 Complexity of multipliers

We describe the complexity of polynomial multiplications (without reduction) by the exact number of the Boolean XOR and AND (#XOR, #AND) operations of two 1-bit operands. This corresponds to the number of XOR- and AND-gates of multipliers. The exact gate complexity (GC) of a certain multiplication method (MM) GC^{MM} for k-bit operands can be expressed by a tuple as follows:

$$GC_k^{MM} = (\#AND, \#XOR) \tag{1}$$

The minimal area and/or energy consumption of a multiplier can be calculated based on its gate complexity and on the area and/or the energy consumption of the used gates ($Area_{AND}, Area_{XOR}$ and E_{AND}, E_{XOR}):

$$\begin{aligned} Area &= \#AND \cdot Area_{AND} + \#XOR \cdot Area_{XOR} \\ Energy &= \#AND \cdot E_{AND} + \#XOR \cdot E_{XOR} \end{aligned} \tag{2}$$

MMs for large k-bit polynomials normally use segmentation of the polynomials into n smaller m-bit terms which are then multiplied. To get the result the partial products are added (i.e. XORed). If this principle of divide and conquer is applied to an k-bit multiplier the resulting ASIC consist of a certain number of m-bit partial multipliers with their own gate complexity GC_m^{MM}:

$$GC_{k=nm}^{MM} = (\#MULT \cdot GC_m^{MM}, \#XOR) \tag{3}$$

The knowledge of the gate complexity of each partial multiplier allows to calculate the gate complexity of full k-bits multipliers. The number of partial multiplications $\#MULT$ and the number of XOR-gates depend on the selected multiplication method and on the segmentation of the operands. Each partial multiplication can be implemented by any MM or even by any combination of MMs. In order to optimize the complexity of a polynomial multiplier it is necessary to determine the optimal combination of different MMs. Formula (3) shows the assessment function that allows to compare different MMs given the fact that the segmentation and the type of the used partial multipliers are the same.

We determined the gate complexity according to (3) for the following MMs: classical MM, Karatsuba MM by segmentation of operands into 4 terms, Montgomery multiplication formulae for 5-, 6- and 7-term operands [3] and the generalized Karatsuba (genKar) algorithm [4]. For each of these MMs we determined the optimized processing sequence (Proc.Seq) and its gate complexity. Only for the classical MM the processing sequence cannot be optimized. For the Karatsuba MM with segmentation of operands into 4 terms we use the optimized processing sequence presented by us in [5]. Due to the lack of space, we cannot explain how the optimized processing sequences can be obtained. Here we give here only their gate complexity, that we use in Algorithm 1 to find the optimal combination of MMs. Table 1 shows the exact complexity of these MMs, with and without using the pre-defined optimized processing sequences.

Table 1. Gate Complexity of investigated MMs

n	**MM**	$\#MULT$	$\#XOR$**, original MM**	$\#XOR$**, MM with Proc.Seq**
4	Karatsuba	9	$40m - 16$	$34m - 11$
5	Montgomery	13	$94m - 40$	$66m - 23$
6	Montgomery	17	$130m - 57$	$96m - 34$
7	Montgomery	22	$184m - 80$	$133m - 47$
n	classical	n^2	$2mn(n-1) - n^2 + 1$	$2mn(n-1) - n^2 + 1$
n	gen.Kar.	$\frac{n^2+n}{2}$	$4mn(n-1) - \frac{3n^2-n}{2} + 1$	$m(2n^2 + n - 3) - \frac{n^2+n}{2}$

Designing an optimal k-bit multiplier requires to know the exact complexity of potential optimal m-bit partial multipliers. The same holds true for optimizing the m-bit partial multiplier. So, to determine the optimal combination of MMs Algorithm 1 is starting from 1-bit polynomials to up to k-bit polynomials. It is essential to determine all possible segmentations for each length of polynomials i, $1 < i \leq k$. The exact complexity of i-bit multipliers is calculated for each MM and all segmentations of i. In the following processing steps in Algorithm 1 the optimal i-bit multiplier is used as the optimal m-bit partial multiplier.

While the algorithm itself is technology independent, there are two technology dependent parameters to be considered. Technology dependent values such as $Area_{AND}, Area_{XOR}$ (or E_{AND}, E_{XOR} respectively) are input variables. Depending on the optimization goal - area or energy - we use respective parts of eq. (2).

Algorithm 1

```
**Input** : $Area_{AND}, Area_{XOR} // if\ optimization\ parameter\ is\ Area$
        $MM = \{MM_{clas}, MM_2, MM_3, ...\} // set\ of\ MMs\ with\ Proc.Seq$
**Initialization** : $MM_{opt}(1) = MM_{clas}(1);\ \ MM_{opt}(i) = empty, 2 \leq i \leq k$
**Calculation** :
  $for\ 2 \leq i \leq k // all\ operands\ of\ smaller\ length$
    $for\ n|2 \leq n \leq i, n\ divides\ i // all\ possible\ segmentations$
      $for\ each\ element\ MM_j\ from\ MM$
        $calculate\ Area(GC_{i=nm}^{MM_j}) // see$ (1), (2), (3) $and$ **Table 1**
        $if\ MM_{opt}(i) = empty\ or\ Area(GC_{i=nm}^{MM_j}) < Area(GC_i^{MM_{opt}(i)})$
          $MM_{opt}(i) = MM_j$
        $end\ if$
      $end\ for$
    $end\ for$
    $for\ s|i > s > 0 // all\ operands\ of\ smaller\ length$
      $if\ \ Area(GC_s^{MM_{opt}(s)}) > Area(GC_{s+1}^{MM_{opt}(s+1)})$
          $MM_{opt}(s) = MM_{opt}(s+1)$
      $end\ if$
    $end\ for$
  $end\ for$
```

3 Evaluation of the optimization results

In order to benchmark our results we are using results from [1] and [4][1]. Since we are mainly interested in ECC we reconstructed data from [1] and [4] to get results for polynomials with a length up to 600 bit[2]. Table 2 shows the gate complexity and calculated area of multipliers for all three approaches. Please note that the number of AND- and XOR-gates of our combinations of MMs are selected based on the results of Algoritm 1, i.e. they reflect the number of gates for the smallest polynomial multipliers for the IHP technology [6]. When comparing the results it becames apparent that the number of AND-gates is smallest for [4]. But the our approach and approach from [1] require by far less XOR-gates. This is the reason for the much smaller area of our multiplier.

[1] by results of [4] we denote those provided for the recursively applied generalized Karatsuba MM

[2] the reconstructed data for polynomials up to 128 bits are the same as those given in [1] and [4]

Table 2. Gate complexity of polynomial multipliers

k,	**reconstructed from [4]**			**reconstructed from [1]**			**our combination of MMs with Proc.Seq**		
bit	$\#AND$	$\#XOR$	area, mm^2	$\#AND$	$\#XOR$	area, mm^2	$\#AND$	$\#XOR$	area, mm^2
163	4536	23417	0.3516	7938	12820	0.2365	7938	11751	0.2221
233	6561	37320	0.5549	12150	23468	0.4137	12150	21066	0.3814
283	8748	48485	0.7227	13122	34108	0.5646	13122	30091	0.5106
409	17496	98039	1.4598	26244	67420	1.1186	29700	54418	0.9716
571	26244	147755	2.1991	39366	104704	1.7259	37179	93383	1.5560

4 Conclusion

In this paper we have presented an algorithm that determines the optimal segmentation of operands and an optimal combination of multiplication methods from a predefined set of MMs. To the best of our knowledge we are the first authors that use MMs which have been optimized to reduce the number of XOR operations, as starting set of such an algorithm. The area of the MM combinations selected by our algorithm is in average about 10% and over 30% smaller than the results presented in [1] and [4], respectively.

References

1. Von zur Gathen, J., Shokrollahi, J.: Efficient FPGA-based Karatsuba multipliers for polynomials over F2. In: Proc. of Selected Areas in Cryptography - SAC 2005, LNCS 3897, pp.359-369, Springer-Verlag, Kingston, ON, Canada (2005)
2. Dyka, Z., Langendoerfer, P.: Area efficient hardware implementation of elliptic curve cryptography by iteratively applying Karatsubas method. In: Proc. of the Design, Automation and Test in Europe Conference and Exhibition, Vol.3, pp.70-75, (2005)
3. Montgomery, P.L.: Five, Six, and Seven-Term Karatsuba-Like Formulae. IEEE Transactions on Computers, vol. 54, no.3, 362–369 (2005)
4. Weimerskirch, A., Paar, C.: Generalizations of the Karatsuba Algorithm for Efficient Implementations. Report 2006/224, Cryptology ePrint Archive (2006), `http://eprint.iacr.org/2006/224.pdf`
5. Peter, S., Langendoerfer, P.: An Efficient Polynomial Multiplier $GF(2^m)$ and its Application to ECC Designs. In: Proc. of the Design, Automation and Test in Europe Conference and Exhibition, pp.1253-1258, (2007)
6. Innovations for High Performance Microelectronics, `http://www.ihp-microelectronics.com/`

Efficient implementation of code-based identification schemes

Pierre-Louis Cayrel, Sidi Mohamed El Yousfi Alaoui, Felix Günther, Gerhard Hoffmann and Holger Rother

CASED – Center for Advanced Security Research Darmstadt,
Mornewegstrasse, 32
64293 Darmstadt
Germany

Abstract. In this paper we present efficient implementations of several code-based identification schemes, namely the Stern scheme, the Véron scheme and the Cayrel-Véron-El Yousfi scheme. For a security of 80 bits, we obtain a signature in respectively 1.048 ms, 0.987 ms and 0.594 ms.

Keywords: Cryptography, Zero-knowledge identification, coding theory, efficient implementation.

1 Introduction

Code-based zero-knowledge identification and signature schemes are an interesting alternative to classical (number theory based) digital signatures. Supposed to resist quantum attacks, several code-based cryptosystems have been developed recently. Shor has showed a quantum algorithm which solves in polynomial time the problems of discrete logarithm and factorization in [9], but no quantum attack exists, so far, to solve the hard problems on which the code-based cryptosystems are based.

In 1993, Stern proposed in [11] the first zero-knowledge identification scheme based on the hardness of the binary syndrome decoding problem. A few years later, Véron in [12] has designed a scheme with a lower communication cost. Recently, Cayrel et al. in [3] have designed a scheme which reduce even more this communication cost.

Using quasi-cyclic and quasi-dyadic constructions, several new constructions like [1, 7] permits to reduce the size of the public matrices. We can use the same kind of matrices in the context of zero-knowledge identification and signature without lower the security of the resulting schemes.

Our contribution In this paper we provide, to our knowledge the first, efficient implementations of the previous schemes for identification and signature. In [2], the authors presented a smart implementation of the Stern scheme but it was more a proof of concept than an efficient implementation.

Organization of the paper Section 2 describes the Stern, Véron and Cayrel-Véron-ElYousfi schemes. Section 3 describes the results of our implementations. Section 4 concludes the paper.

2 Code-based zero-knowledge identification schemes

In code-based cryptography, there have been many attempts to design identification schemes. In such constructions, there are two main goals: On the one hand, a prover wants to convince a verifier of its identity. On the other hand, the prover does not want to reveal any additional information that might be used by an impersonator. In the following, we will give an overview of three proposals in this area.

2.1 Stern scheme

The first code-based zero-knowledge identification scheme was presented at Crypto'93 by Stern [11], its security is based on the syndrome decoding (SD) problem. It uses a public parity-check matrix of the code over the binary field $\mathbb{F}_2$. This scheme is a multiple-rounds identification protocol, where each round is a three-pass interaction between the prover and the verifier. A cheater has a probability of 2/3 per round to succeed in the protocol without the knowledge of the secret key. The number of rounds depends on the security level needed; for 80 bits security level, one needs about 150 rounds. For instance to achieve the weak and strong authentication probabilities of 2^{-16} and 2^{-32} according the norm ISO/IEC-9798-5, one needs respectively 28 and 56 rounds.

2.2 Véron scheme

In 1996, Véron proposed in [12] a dual version of Stern's scheme. It uses a generator matrix instead of a parity-check matrix of the code, which has the advantage to reduce slightly the communication costs. Véron's scheme, as Stern's, is a multiple rounds zero-knowledge protocol, where each round is a three-pass interaction between the prover and the verifier, for which the success probability for a cheater is 2/3. Moreover, Véron suggested in [12] to use special techniques over finite fields to reduce the computation and storage complexity of his scheme.

2.3 Cayrel-Véron-El Yousfi scheme

In 2010, Cayrel, Véron, and El Yousfi (CVE) presented in [3] a five-pass identification protocol using q-ary codes instead of binary codes. In addition to the new way to calculate the commitments, the idea of this protocol uses another improvement which is inspired by [8, 10]. The main achievement of this proposal is to decrease the cheating probability of each round from 2/3 for Stern's and Véron's schemes to 1/2. This allows to decrease the communication complexity and then to provide the desired security level in fewer rounds compared to Stern and Véron constructions. Furthermore, this scheme offers a small public key size, about 4 kBytes, whereas that of Stern and Véron scheme is almost 15 kBytes for the same level of security. It is proven in [3] that this scheme verifies the zero-knowledge proof and its security is based on the hardness of the syndrome decoding problem defined over $\mathbb{F}_q$.

Since a large public matrix size is one of the drawbacks of code-based cryptography, there have been many proposals which consists of replacing the random codes by particular structured codes, namely quasi-cyclic proposed by Gaborit and Girault in [5] or quasi-dyadic codes proposed by Miscozki and Barreto in [7]. We can use the both variants in the three identifications schemes presented above, in order to store the public matrix more efficiently.

We can also mention that the three presented identification schemes can be turned into secure signature schemes by using the idea of Fiat-shamir paradigm.

3 Efficient implementation

3.1 Description

In total, six different schemes have been implemented in C: the Stern, Véron and CVE identification schemes and the corresponding signature schemes based on the Fiat-Shamir transform [6]. The idea of the transform is to split the identification scheme in two parts. In the first part, the signer runs the identication scheme as before, but only recording the responses without any checks. In the second part, the verifier replays the saved responses and performs the necessary checks. This also explains the relatively high signature size of schemes based on the Fiat-Shamir transform. It also shows the varying sizes of the signatures, as the given responses change from run to run with high probability.

All implementations use the SHA-3 finalist Keccak [4], both as hash function and as random oracle. All tests have been carried out on an Intel(R) Core(TM)2 Duo CPU E8400@3.00GHz machine, the source code is publicly available.[1].

3.2 Results

The following tables give the timings of some test runs. For the signature schemes files of size about 1 MB, 10 MB and 25 MB have been used. As expected, the actual responses (i.e. challenges) vary from run to run. The signatures sizes in Table 2 are taken as approximate values across multiple runs.

As the code of the implementations does not use any object-oriented features, a straightforward efficient Java implementation should be possible as well.

[1] `http://cayrel.net/spip.php?article189`

	Stern	**Véron**	**CVE**
Rounds	28	28	16
n, r, w	768, 384, 76	768, 384, 76	144, 72, 55
Security level	2^{80}	2^{80}	2^{80}
Random	1.048 ms	0.987 ms	0.594 ms
n, r, w	1024, 512, 128	1024, 512, 128	256, 128, 97, 256
Security level	2^{73}	2^{73}	2^{143}
Quasi-Cyclic	1.893 ms	1.634 ms	1.829 ms
Quasi-Dyadic	2.655 ms	2.522 ms	1.775 ms

Table 1. Timing results for Stern, Véron and Cayrel, Véron, and El Yousfi (CVE) identification schemes.

	Stern	**Véron**	**CVE**
Rounds	28	28	16
n, r, w	768, 384, 76	768, 384, 76	144, 72, 55
Security level	2^{80}	2^{80}	2^{80}
Message size [by.]	**Random (Sign/Verify [ms])**		
1.363.024	0.008/0.007	0.008/0.007	0.013/0.012
10.317.040	0.055/0.054	0.054/0.054	0.106/0.118
23.766.127	0.126/0.125	0.124/0.124	0.247/0.243
Signature size [by.]	60.000	60.000	15.000
n, r, w	1024, 512, 128	1024, 512, 128	256, 128, 97, 256
Security level	2^{73}	2^{73}	2^{143}
Message size [by.]	**Quasi-Cyclic (Sign/Verify [ms])**		
1.363.024	0.008/0.007	0.008/0.007	0.014/0.013
10.317.040	0.056/0.053	0.055/0.054	0.108/0.105
23.766.127	0.129/0.126	0.125/0.125	0.247/0.243
Message size [by.]	**Quasi-Dyadic (Sign/Verify [ms])**		
1.363.024	0.009/0.008	0.009/0.008	0.014/0.013
10.317.040	0.056/0.054	0.056/0.055	0.104/0.108
23.766.127	0.127/0.125	0.126/0.126	0.247/0.243
Signature size [by.]	80.000	80.000	25.000

Table 2. Timing results for Stern, Véron and Cayrel, Véron, and El Yousfi (CVE) signature schemes. The signature sizes are approximate values.

4 Conclusion

In this paper, we have described three existing code-based identification and signature and have provided a detailed comparison of their implementation. As a result, we obtain three very fast signature (in less than 1ms) but very long signature size from 25.000 for CVE to 80.000 bytes for Stern and Véron. The security of the implementations faces side-channel attacks (like SAP and first order DPA) has been studied in [2] but the security of those implementations faces fault-injection or higher order DPA has not been studied yet. The source codes are available here : `http://cayrel.net/spip.php?article199`.

References

1. T. P. Berger, P.-L. Cayrel, P. Gaborit, and A. Otmani. Reducing Key Length of the McEliece Cryptosystem. In *Progress in Cryptology – Africacrypt'2009*, volume 5580 of *LNCS*, pages 77–97. Springer, 2009.
2. P.-L. Cayrel, P. Gaborit, and E. Prouff. Secure Implementation of the Stern Authentication and Signature Schemes for Low-Resource Devices. *CARDIS*, 2008.
3. P.-L. Cayrel, P. Véron, and S. M. Y. Alaoui. A Zero-Knowledge Identification Scheme Based on the q-ary Syndrome Decoding Problem. In *Selected Areas in Cryptography*, pages 171–186, 2010.
4. G. Bertoni, J. Daemen, M. Peeters and G. V. Assche. The Keccak sponge function family. `http://keccak.noekeon.org/`.
5. P. Gaborit and M. Girault. Lightweight Code-based Authentication and Signature. In *IEEE International Symposium on Information Theory – ISIT'2007*, pages 191–195, Nice, France, 2007. IEEE.
6. M. Abdalla and J.H. An and M. Bellare and C. Namprempre. From Identification to Signatures Via the Fiat-Shamir Transform: Necessary and Sufficient Conditions for Security and Forward-Security. *IEEE Transactions on Information Theory*, pages 3631–3646, 2008.
7. R. Misoczki and P. S. L. M. Barreto. Compact McEliece Keys from Goppa Codes. Preprint, 2009. `http://eprint.iacr.org/2009/187.pdf`.
8. A. Shamir. An Efficient Identification Scheme Based on Permuted Kernels. In *Proceedings of the 9th Annual International Cryptology Conference on Advances in Cryptology*, CRYPTO '89, pages 606–609, London, UK, 1990. Springer-Verlag.
9. P. W. Shor. Polynomial-Time Algorithms for Prime Factorization and Discrete Logarithms on a Quantum Computer. *SIAM Journal on Computing*, 26:1484–1509, 1995.
10. J. Stern. Designing Identification Schemes with Keys of Short Size. In *Advances in Cryptology – Proceedings of CRYPTO '94*, volume 839, pages 164–173, 1994.
11. J. Stern. A New Identification Scheme Based on Syndrome Decoding. In *Proceedings of the 13th Annual International Cryptology Conference on Advances in Cryptology*, pages 13–21, New York, NY, USA, 1994. Springer-Verlag New York, Inc.
12. P. Véron. Improved Identification Schemes Based on Error-Correcting Codes. *Appl. Algebra Eng. Commun. Comput.*, 8(1):57–69, 1996.

Improved Software Implementation of DES Using CUDA and OpenCL

D. Noer[*], A.P. Engsig-Karup[*], E. Zenner[†]
Technical University of Denmark, Lyngby, Denmark

This work is concerned with the development of a fast DES bitslice brute-force software tool which utilize consumer Graphics Processing Units(GPUs) and shows improved performance over existing implementations [1]. The use of modern GPUs in high-performance computing is a new trend, where such devices may be useful for offloading computationally intensive tasks for achieving a significant performance boost in comparison with traditional use of general purpose Central Processing Units (CPUs). Programming GPUs are supported by new programming models based on the C language, e.g., the most widely used vendor-specific CUDA [6] and and the industry-wide OpenCL standard [4].

Modern GPUs can be attractive for parallel processing because these architectures by design have hundreds of processing cores and have high on-chip bandwidth close to one order in magnitude larger than modern CPUs. These GPUs have good support for hiding latency in memory transactions through massive multithreading with low context switch overhead. The processing of instructions in the thread contexts is based on the Single Instruction Multiple Data (SIMD) processing paradigm and is therefore suitable for algorithms that can expose a high degree of data parallelism.

The Data Encryption Standard (DES) was chosen as a case study because the block-cipher uses permutations and substitutions of data, rather than the arithmetic calculations which GPUs are known to excel in. The goal is to evaluate the potential of GPUs for this type of application. Furthermore, the DES cipher has a limited 56 bit key space, which has been successfully cracked by [3] on FGPAs. However, FGPAs are much more expensive than GPUs and requires much more programming effort in comparison with CUDA and OpenCL.

A bitsliced [2] implementation of DES was initially considered to be a suitable candidate algorithm for implementation on GPUs. The bitslice method is an emulated SIMD, that utilizes the n-bit registers as a slice of the data vector, making it possible to permute n bits per operation. Our tool is based on highly optimized lookup tables called SubstitutionBOXes (SBOXs) [5]. The nonlinear

[*] Department of Informatics and Mathematical Modelling. The work has been supported by grant no. 09-070032 from the Danish Research Council for Technology and Production Sciences and with resources of GPULAB (http://gpulab.imm.dtu.dk).

[†] Department of Mathematics

SBOXs are converted from a lookup table to pure logic, which on average requires 56 operations. Thus, this is much faster than cutting the distinct key values from the slice and sending them through a lookup table thereby substituting excessive high-latency data transfers with bitwise operations enabling fast processing.

The linear permutations are optimized by finding permutations of permutations and finally reducing them to a single permutation. With permutation reduction in a bitslice implementation, the need to permute data can be circumvented, by letting the ordering of bitslices in the SBOXs function be permuted, which in effect eliminates most movement of data.

An affordable Nvidia GeForce GTX 275 gaming card controlled by the CPU host has been used for the initial development. A naive implementation shows a ten-fold speedup in comparison with the same method running on a Intel CoreI7@2.66GHz CPU. However, this implementation does not utilize the scarce low-latency memory locations (i.e. registers, shared memory and constant memory) on the GPU. Therefore by utilizing these low-latency memories it is possible to reduce the memory fetch time to a fraction, which is found to achieve 13 times the CPU speed.

Further analysis of the model shows that the implementation uses more registers than can be made available per thread on the Nvidia GPUs. The implementation was improved by hard coding static parts of the model thereby reducing the registers per thread to below the hardware limit of 127 registers per thread. This improvement let the entire model rely on using the fast registers, resulting in 18 times the CPU speed. With a number of minor additional improvements the current model achieves 20 times speedup compared to the CPU.

We find that the resulting implementation is able to search up to 680 million keys/s on a GTX 275, which approximately doubles the performance in comparison with previous work [1] on a similar architecture. The major difference between the two models is found in the handling of the bitslices. The model described in [1] relies on precomputed bitslices fetched from constant memory. In the present work, the model programmed in CUDA calculate all bitslices at runtime, which is faster than fetching them from memory because the entire model now fits in the registers.

Further performance improvements will be pursued on AMD GPUs which will requires an implementation of the model in OpenCL. Such an implementation will also be able to execute on general heterogenous hardware setups (including Nvidia GPUs) and can therefore be subject to additional investigations in performance comparison and differences. These findings together with latest results will be presented at the conference.

We remark that a perspective in this work is that this type of DES bruteforcer can be distributed over any number of GPUs, thus supplying organized groups the power of a super computer. It is doubtful that any organization have the computing power to exhaust the key space of modern ciphers, but relative short password searches could theoretically be conducted successfully on distributed GPUs.

References

[1] Giovanni Agosta, Alessandro Barenghi, Fabrizio De Santis, and Gerardo Pelosi. Record setting software implementation of des using cuda. In *Proceedings of the 2010 Seventh International Conference on Information Technology: New Generations*, ITNG '10, pages 748–755, Washington, DC, USA, 2010. IEEE Computer Society.

[2] Eli Biham. A Fast New DES Implementation in Software. In *Proceedings of the 4th International Workshop on Fast Software Encryption*, FSE '97, pages 260–272, London, UK, 1997. Springer-Verlag.

[3] Electronic Foundation. *Cracking DES*. O'Reilly Media, 1998.

[4] Khronos Group. The OpenCL specification V1.0.48 , 2010.

[5] Matthew Kwan. Reducing the Gate Count of Bitslice DES, October, 2000. IACR Eprint archive.

[6] NVIDIA. NVIDIA CUDA C Programming Guide Version 3.2, 2010.

Full Lattice Basis Reduction on Graphics Cards

Timo Bartkewitz[1] and Tim Güneysu[2]

[1]Department of Computer Science
Bonn-Rhine-Sieg University of Applied Sciences, Germany
timo.bartkewitz@h-brs.de
[2]Horst Görtz Institute for IT Security
Ruhr-University Bochum, Germany
gueneysu@crypto.rub.de

Abstract. We present the first implementation of a full lattice basis reduction for graphics cards. In this work, we show that graphics cards are well suited to apply alternative algorithms that were merely of theoretical interest so far due to their enormous demand for computational resources and requirements on parallel processing. We modified and optimized these algorithms to fit the architecture of graphics cards, in particular we focused on Givens Rotations and the All-swap reduction method. Eventually, our GPU implementation achieved a significant speed-up for given lattice compared to the NTL implementation running on an CPU (e.g., a speed-up factor of 13.7 for the same financial investment), providing at least the same reduction quality.

Key words: Lattice Basis Reduction, SBP, Parallelization, Givens Rotations, All-Swap Algorithm, CUDA

1 Introduction

The lattice basis reduction is an important and interesting tool in linear algebra. Various applications concern the factorization of polynomials and integer numbers as well as solving knapsack, hidden number problems, and many more problems [4, 5] – all enabled by finding a relatively short lattice basis (*Shortest Basis Problem* or *SBP*) and the shortest vector for a given lattice (*Shortest Vector Problem* or *SVP*). In particular, the latter method could also be used to break instances of the RSA public-key cryptosystem [7]. Beside this factoring-based cryptosystem, there is the class of lattice-based cryptosystems that are assumed to be secure against attacks with quantum computers. Thus, the performance of lattice basis reduction is indeed essential for reasonable security estimations.

Our contribution: In this paper, we will adopt and improve parallel algorithms for lattice basis reduction to achieve optimal results on graphics cards. A major part of a LLL-based lattice basis reduction algorithm is the orthogonalization.

2 Previous Work

Shortest lattice vector enumeration, virtually the main part of the BKZ which targets the approximated SVP, was ported to GPUs [3]. But in order to obtain reasonable advances in performance it still requires a strong pre-reduction of the lattice basis and hence a fast LLL-algorithm.

3 Preliminaries

In linear algebra, a lattice in $\mathbb{R}^n$ is a discrete, additive, Abelian subgroup of $\mathbb{R}^n$ consisting of points.

Definition 1 (Lattice). *Let $b_1, b_2, \dots, b_k \in \mathbb{R}^d, k \leq d$ linear independent, the set*

$$\mathcal{L} = \left\{ u \in \mathbb{R}^d | u = \sum_{i=1}^{k} a_i b_i, a_i \in \mathbb{Z} \right\}$$

is called a lattice.

Every lattice $\mathcal{L}$ can be represented by a set $\mathcal{B} = \{b_1, b_2, \dots, b_k\}$ of column vectors. We call $\mathcal{B}$ the basis of the lattice $\mathcal{L}$, thus $\mathcal{L}(\mathcal{B})$ is the set of all finite, integer linear combinations of the basis vectors b_i.

Detailed information on lattices and their properties can be found in [1].

Lattice Basis Reduction The lattice basis reduction deals with the problem to find a short lattice basis for a given lattice basis (SBP). In practice, finding a shortest vector (SVP) in this basis is of particular importance. In 1982 Lenstra, Lenstra and Lovász proposed the first lattice basis reduction [6] that terminates in polynomial runtime, according to the lattice dimension, which is known as the *LLL-algorithm*.

4 Computations on Graphics Cards

General-purpose computing on graphics processing units (GPGPU) is the shift of computations that are traditionally handled by the *central processing unit* (CPU) or *host* processor, to the *graphics processing unit* (GPU), also known as *device*. In this paper, we focus on nVidia GPUs and CUDA that can be programmed with *C for CUDA*, a C language derivative with special extensions.

5 Lattice Basis Reduction on Graphics Cards

Lattice basis reduction has three main phases: first the *basis orthogonalization* (computation of Gram-Schmidt coefficients), second the *size reduction* of the basis and third the *basis permutation*.

5.1 Parallel Orthogonalization

For parallel orthogonalization the *QR decomposition* is a tool that generates an orthogonal basis. In this paper, we focus on a parallel variant of the Givens rotations. Detailed informations on the QR decomposition and the corresponding methods can be found in [1].

Our approach realizes an effective way to implement the Givens rotations whereas one thread block is responsible for two affected rows implied by a Givens rotation to insert a single zero.

5.2 Parallel Basis Size Reduction

We propose a novel pattern that optimally fits the architecture of graphics cards. However, we decoupled the size reduction of the basis from that of the Gram-Schmidt coefficients. Hence, we first reduce the Gram-Schmidt coefficients and compute the nearest integer at a time.

Next, we reduce the basis with help of the pre-computed Gram-Schmidt coefficients involving a row-wise weighted sum.

5.3 Parallel Basis Permutation

The so-called *All-Swap* [8] lattice basis reduction intends to process as much as possible of the entire basis with respect to orthogonalization, size reduction and permutation by swapping. The algorithm works iteratively in competitive alternating phases, an *odd* and an *even* phase.The original All-Swap floating point algorithm was proposed by Heckler and Thiele [2].

Here, we introduce a variant from that we expect a better reduction quality due to a higher number of swap operations. Instead of swapping two adjoining vectors by which means sorting them according to their squared 2-norms $\|b_i^*\|_2^2$, blocks of size 2^l, with $2^l \leq k$, vectors will be sorted. Our so called *ordered All-swap* approach is represented by Algorithm 1.

The reduction parameter δ' is deduced from the original δ that is included in the LLL-algorithm.

6 Results

For our experiments, we used a nVidia GTX 280 graphics cards with 1 GiB video RAM and an Intel Core 2 Quad running Windows 7 64-bit. The results were obtained using the CUDA toolkit and SDK *3.2* and the CUDA driver *260.89*.

To provide reasonable results for the full lattice basis reduction, we consider random lattice bases , and random lattices bases in Hermite normal form.

Compared to runtime results from NTL[1] that involves the Schnorr-Euchner algorithm in double floating point precision using Givens rotations (`G_LLL_FP()`

[1] All measurements were performed on the same system as presented above.

Algorithm 1 Ordered All-Swap Lattice Basis Reduction

Input: Lattice basis $\mathcal{B} = (b_1, b_2, \ldots, b_k) \in \mathbb{R}^{d \times k}$, reduction parameter δ' with $\delta' > \frac{4}{3}$ and block-size parameter l

Output: δ'-All-swap-reduced basis $\mathcal{B}$

```
$phase = 0$
**while** sorting is possible for any block in phase **do**
  Approximate basis $\mathcal{B}' = (\mathcal{B})'$
  Compute Gram-Schmidt coefficients $\mu_{i,j}$
  Size reduce the basis $\mathcal{B}$
  Split the basis into $m$ blocks of size $2^l$ starting with $b_{1+phase \cdot 2^{l-1}}$
  **for** $i = 1$ to $m$ **parallel do**
    Using an appropriate sorting algorithm to sort the block
    $b_{2^l(i-1)+phase \cdot 2^{l-1}+1}, \ldots, b_{2^l i+phase \cdot 2^{l-1}}$ by its squared 2-norms, i.e.
    $\|b_r^*\|_2^2 \leq \delta' \|b_s^*\|_2^2, \quad 2^l(i-1) + phase \cdot 2^{l-1} + 1 \leq r < s \leq 2^l i + phase \cdot 2^{l-1}$
    $phase = phase \oplus 1$
  **end for parallel**
**end while**
Approximate basis $\mathcal{B}' = (\mathcal{B})'$
Compute Gram-Schmidt coefficients $\mu_{i,j}$
Size reduce the basis $\mathcal{B}$
```

our implementation achieves a speed-up of about 12.5 ($l = 1$) and 18.6 ($l = 2$) on average.

Figure 1 shows the runtime performance for both randomly chosen lattice bases and randomly chosen lattice bases in Hermite normal form using a logarithmic scale.

Normalizing the speed-up of the GPU-based implementation according to the higher cost of its computing system, we still have a 13.7 times higher performance compared to the corresponding CPU-based system for the same amount of financial investment.

7 Conclusion and Future Work

In this paper we presented the first implementation of a full lattice basis reduction on graphics cards. We achieved promising results with respect to other CPU-based implementations, such as NTL. We introduced a variant of the All-swap algorithm that delivers better reduction results with decreased runtime with respect to given lattice bases. Our implementation can also be used to find short vectors, however at the cost of a higher runtime.

Future work involves the OpenCL framework that offers quadruple floating point precision in the next versions. Thus, it would become possible to reduce either lattice bases with very high dimensions or lattice bases consisting of large entries which is, as of now, restricted by the current GPU generation. Alter-

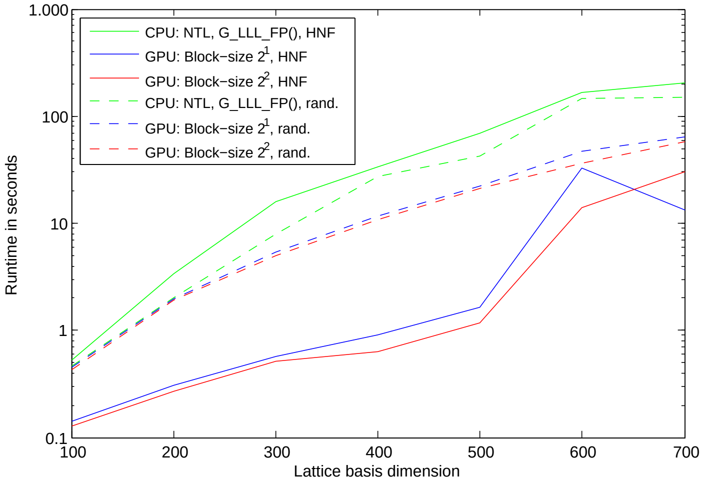


Fig. 1. Runtime for random lattice bases and random lattice bases in Hermite normal form with $\delta' = 1.34$ ($l = 1$), $\delta' = 1.55$ ($l = 2$) and $\delta = 0.99$ (NTL)

natively, future work could apply the proposed approach as a pre-reduction for lattice enumeration.

References

1. J.E. Gentle, W. Hrdle, and Y. Mori. *Handbook of Computational Statistics*. Springer-Verlag Berlin Heidelberg, 2004.
2. C. Heckler and L. Thiele. Parallel Complexitiy of Lattice Basis Reduction and a Floating-Point Parallel Algorithm. In *PARLE*, pages 744–747, 1993.
3. Jens Hermans, Michael Schneider, Johannes Buchmann, Frederik Vercauteren, and Bart Preneel. Parallel shortest lattice vector enumeration on graphics cards. In *AFRICACRYPT*, pages 52–68, 2010.
4. M.J. Hinek. Lattice Attacks in Cryptography: A Partial Overview. Technical report, School of Computer Science, University of Waterloo, 2004.
5. A. Joux and J. Stern. Lattice Reduction: a Toolbox for the Cryptanalyst. *Journal of Cryptology*, 11:161–185, 1994.
6. A.K. Lenstra, H.W. Lenstra, and L. Lovász. Factoring polynomials with rational coefficients. *Mathematische Annalen*, Volume 261(4):515–534, 1982.
7. R.L. Rivest, A. Shamir, and L. Adleman. A Method for Obtaining Digital Signatures and Public-Key Cryptosystems. *Communications of the ACM*, 21:120–126, 1978.
8. G. Villard. Parallel lattice basis reduction. In *ISSAC '92: Papers from the international symposium on Symbolic and algebraic computation*, pages 269–277, New York, NY, USA, 1992. ACM.

Intractability of a Linear Diophantine Equation Discrete Log Problem based Asymmetric Cryptosystem

M.R.K.Ariffin[1,2] and N.A.Abu[1,3]

[1] Al-Kindi Cryptography Research Laboratory, Institute for Mathematical Research, Universiti Putra Malaysia (UPM), Selangor, Malaysia
[2] Department of Mathematics, Faculty of Science, Universiti Putra Malaysia (UPM)
[3] Faculty of Information Technology and Communication, Unitversiti Teknikal Malaysia (UTeM), Melaka, Malaysia

1 Introduction

Efficient implementation of cryptographic primitives either on hardware or software gave rise to "cryptographic efficiency" issues. Issues surround the key length, speed and computational effort to execute. Suggestions have been made that ECC should be the preferred asymmetric cryptosystem when compared to RSA since it provides security with shorter keys [8]. However, in certain situations where a large block needs to be encrypted, RSA is the better option than ECC because ECC would need more computational effort to undergo such a task since ECC is "computational intensive" [7]. In 1998 the cryptographic scheme known as NTRU was proposed with better "cryptographic efficiency" when compared to RSA and ECC [6]. Much research has been done to push NTRU to the forefront [5]. However, this paper would focus on making comparisons against discrete log based cryptosystems and elliptic curve based cryptosystems only.
The cryptographic scheme in this paper is based on what the authors define as the Linear Diophantine Equation Discrete Log Problem (LDEDLP). It is not the intention to go into "provable" security concepts in this initial stage. The immediate objective is to be able to set up a mathematical concept that exhibits one-way characteristic functionality and differs from conventional "one-way" mathematical concepts (i.e. discrete log problem, integer factorization, elliptic curves etc.). The reason? For better "cryptographic efficiency". Specifically, the LDEDLP arises within the scheme when one attempts to solve the matrix decomposition problem. The ability to re-produce the corresponding two square matrices from its product where both matrices are private and one of them is singular is related to solving the LDEDLP (albeit in a stronger setting when compared to the situation where certain parameters are known). The authors propose that the LDEDLP as outlined in this paper is also another discrete log problem that has secure cryptographic qualities coupled with the above described "cryptographic efficiency" qualities. The intractability of the LDEDLP will be presented. Results relating the LDEDLP to the Diffie Hellman key exchange and the RSA cryptosystem will be discussed. Next, the AA$_\beta$-cryptosystem which is

based upon the LDEDLP will be introduced. The AA$_\beta$-cryptosystem transmits a ciphertext consisting of three parameters and utilizes only the multiplication operation for encryption and decryption. Finally, we will conclude by comparing “cryptographic efficiency” characteristics of the AA$_\beta$-cryptosystem with RSA and ECC cryptographic schemes.

2 The linear diophantine equation discrete log problem (LDEDLP)

The LDEDLP is based upon the linear diophantine equation which is of the form $U = Vx + Wy$. The following definitions would give a precise idea regarding the LDEDLP.

Definition 1. *Let $U = Vx^* + Wy^*$. We define the pair (x^*, y^*) as the preferred integers used to obtain U. The pair (x^*, y^*) is an element from the set of solutions of $U = Vx + Wy$ which contains infinitely many elements.*

Definition 2. *The linear Diophantine equation given by $U = Vx + Wy$ is defined to be prf-solved when (x^*, y^*) are found in order to obtain U. The LDEDLP is solved when $U = Vx + Wy$ is prf-solved.*

Remark 1. If the solution set of an equation is restricted to a finite set or can be limited to a finite number of possibilities the solution set can be found by brute force, that is, by testing each of the possible values.

Lemma 1. *The linear Diophantine equation $U = Vx^* + Wy^*$ is computationally infeasible to be prf-solved by brute force.*

Definition 3. *(DH-Diophantine equation) From the Diffie Hellman key exchange procedure ($g^a \equiv A(modp)$) we define the DH-Diophantine equation as $A = g^a - pt$ for $t \in \mathbb{Z}$.*

Definition 4. *(RSA-Diophantine equation) From the RSA encryption procedure ($C \equiv M^e(modN)$) we define the RSA-Diophantine equation as $C = M^e - Nt$ for $t \in \mathbb{Z}$.*

Proposition 1. *If the linear Diophantine equation $U = Vx + Wy$ is computationally feasible to be prf-solved then both the DH-Diophantine equation and RSA-Diophantine equation are computationally feasible to be prf-solved.*

3 The AA$_\beta$-Cryptosystem

The AA$_\beta$-cryptosystem is based upon the AA$_\beta$-function which was first introduced by Ariffin and Abu in 2009 [1]. It was cryptanalyzed by Blackburn in 2010 [3]. In this work we incorporate the AA$_\beta$-matrices that were introduced by Blackburn in his cryptanalysis. It has to be mentioned that the success of the attack was not due to the AA$_\beta$-function but was due to weaknesses in the design of the public key. An attempt to strengthen against the attack was disclosed by Ariffin et. al. in 2010 [2]. However, it was not successful. In this work we re-examined Blackburn's attack and exploited the AA$_\beta$-matrices to strengthen the cryptosystem. We state here the definition of the AA$_\beta$-function together with other definitions in relation to it.

Definition 5. *The set of binary strings with length of k bits is defined by* $S^k = \left\{ s = \{b_i\}_{i=0}^{k-1} \,|b_i \in \{0,1\} \right\}$ *where* $k \in \mathbb{Z}^+$.

Definition 6. *Let* $\alpha, \beta \in \mathbb{Z}^+$ *and* $\alpha < \beta$ *and both are integers of k-bit length. The* AA_β*-function is defined as*

$$AA_\beta(x_i) = \begin{cases} (\alpha x_{i-1} + x_i) \text{ if } \; b_i = 0 \\ (x_{i-1} + \beta x_i) \text{ if } \; b_i = 1 \end{cases}$$

where $i = 0, 1, 2, \ldots k-1$ $x_{-1} = 0$ $x_0 \in \mathbb{Z}^+$ *and* $s \in S^k$.

Lemma 2. *[1] Let* $s \in S^k$ *and* AA_β *a function as defined in Definition 6, let* $G = x_0 \in \mathbb{Z}^+$ *be a generator then* $AA_\beta^s(G) = AA_\beta^s(1) \cdot G = mG$ *where* $m \in \mathbb{Z}^+$. *The integer generated by A will be denoted* m_A *while the integer generated by B will be denoted* m_B.

3.1 The AA$_\beta$ - matrices

In 2010, Blackburn identified another mechanism to construct either the integer m_A or m_B when conducting the attack upon the AA$_\beta$-cryptosystem then. Let the integer matrices (to be known as the AA$_\beta$-matrices) be identified as follows:

$$\mathbf{A}_0 = \begin{pmatrix} 1 & \alpha \\ 1 & 0 \end{pmatrix}, \mathbf{A}_1 = \begin{pmatrix} \beta & 1 \\ 1 & 0 \end{pmatrix}$$

Correspondent A (Along) and B (Busu) will generate their private strings $d_A, d_B \in S^k$ respectively. Along will compute the integer matrix given by

$$\mathbf{A} = \mathbf{A}_{b_{k-1}} \mathbf{A}_{b_{k-2}} \cdots \mathbf{A}_0$$

where the choice of the matrix $\mathbf{A}_{b_j}$ to be utilized depends on the binary element. If the binary is 0 choose $\mathbf{A}_0$ otherwise choose $\mathbf{A}_1$. The integer m_A is the top left entry of the resulting matrix. Busu will also compute his m_B in the same manner (Busu's resultant integer matrix will be denoted as $\mathbf{B}$).

We will now define parameters needed for the renewed AA_β-cryptosystem.

Definition 7. *The private ephemeral base matrix* $\boldsymbol{G}_{A1}$ *is a 2 X 2 singular matrix. We will denote the integers within the matrix as follows:*

$$\boldsymbol{G}_{A1} = \begin{pmatrix} gA1_{11} & gA1_{12} \\ gA1_{21} & gA1_{22} \end{pmatrix}$$

Each element will be chosen of "minimum length" (i.e. as long as $\boldsymbol{G}_{A1}$ is a singular matrix). There will be three base matrices namely $\boldsymbol{G}_{A1}$,$\boldsymbol{G}_{A2}$ and $\boldsymbol{A}_2$.

Definition 8. *The ephemeral private seed is a k-bit random binary strings $d_A \in S^k$.*

Definition 9. *The ephemeral private key is the 2 X 2 non-singular matrix $\boldsymbol{A}$, and its inverse $\boldsymbol{A}^{-1}$. For efficient implementation integers within the matrix $\boldsymbol{A}$, are reduced in size by taking only the first 2k-bits of the resulting integer. We will denote the integers within the matrix as follows:*

$$\boldsymbol{A}_1 = \begin{pmatrix} a1_{11} \; a1_{12} \\ a1_{21} \; a1_{22} \end{pmatrix}$$

Observe that, differing from the work by Ariffin and Abu in 2009, instead of just utilizing $a1_{11}$ as the private key, we know utilize the whole resulting matrix.

Definition 10. *Along's public key, are the matrices $\boldsymbol{E}_{1A}, \boldsymbol{E}_{2A}, \boldsymbol{E}_{3A}$ and $\boldsymbol{E}_{4A}$ defined by:*

$$\boldsymbol{E}_{1A} = \boldsymbol{G}_{A1}\boldsymbol{A}_1 \tag{1}$$

$$\boldsymbol{E}_{2A} = \boldsymbol{G}_{A1}\boldsymbol{A}_2 \tag{2}$$

$$\boldsymbol{E}_{3A} = \boldsymbol{G}_{A2}\boldsymbol{A}_1 \tag{3}$$

$$\boldsymbol{E}_{4A} = \boldsymbol{A}_1^{-1}\boldsymbol{A}_2\boldsymbol{A}_1 \tag{4}$$

Observe that the matrices (1)-(4) are all singular. The maximum length of an element in $\mathbf{E}_{1A}$ is $(2k + l_1)$-bits, where l_1 is the length of largest value in $\mathbf{G}_{A1}$, the maximum length in $\mathbf{E}_{2A}$ is $(l_1 + l_2)$-bits, where l_2 is the length of largest value in $\mathbf{A}_2$, the maximum length in $\mathbf{E}_{3A}$ is $(2k + l_3)$-bits, where l_3 is the length of largest value in $\mathbf{G}_{A2}$ and the maximum length in $\mathbf{E}_{4A}$ is $(3k + l_2)$-bits.

Definition 11. *Along's corresponding private keys are given by:*

$$\boldsymbol{D}_{1A} = \boldsymbol{A}_1^{-1}\boldsymbol{A}_2 \tag{5}$$

$$\boldsymbol{d}_A = \boldsymbol{C}_3\boldsymbol{A}_1^{-1} \tag{6}$$

$$\boldsymbol{D}_{2A} = (\boldsymbol{E}_{3A} - \boldsymbol{d}_A)^{-1} \tag{7}$$

The parameters within $\boldsymbol{D}_{2A}$ must be chosen such that the inverse exists.

Definition 12. *Suppose Along is sending a plaintext to Busu utilizing Busu's public keys $\boldsymbol{E}_{1B}, \boldsymbol{E}_{2B}, \boldsymbol{E}_{3B}$ and $\boldsymbol{E}_{4B}$. Let $\boldsymbol{M}$ be Along's plaintext and $\boldsymbol{K}_A$ be Along's ephemeral session key. Both $\boldsymbol{M}$ and $\boldsymbol{K}_A$ are arbitrary 2 X 2 matrices. Let $\boldsymbol{C}_1 = \boldsymbol{K}_A\boldsymbol{E}_{1B} + \boldsymbol{M}\boldsymbol{K}_A$, $\boldsymbol{C}_2 = \boldsymbol{K}_A\boldsymbol{E}_{2B} + \boldsymbol{M}\boldsymbol{E}_{3B}$ and $\boldsymbol{C}_3 = \boldsymbol{K}_A\boldsymbol{E}_{4B}$ be the ciphertexts that Along will relay to Busu.*

Proposition 2. $[\boldsymbol{C}_2 - (\boldsymbol{C}_1\boldsymbol{D}_{1B})]\boldsymbol{D}_{2B} = \boldsymbol{M}$

3.2 The AA$_\beta$ - public key cryptography scheme

The scenario is that Along will send an encrypted message to Busu. Busu will provide Along with his public key pair $\mathbf{E}_{1B}, \mathbf{E}_{2B}, \mathbf{E}_{3B}$ and $\mathbf{E}_{4B}$. Along will then generate an ephemeral session key $\mathbf{K}_A$ according to Definition 12. Along will proceed to generate $\mathbf{M}$ and then compute $\mathbf{C}_1$, $\mathbf{C}_2$ and $\mathbf{C}_3$. Then Along transmits the three-parameter ciphertext $(\mathbf{C}_1, \mathbf{C}_2, \mathbf{C}_3)$ to Busu. Upon receiving the ciphertext from Along, Busu will proceed to compute $[\mathbf{C}_2 - (\mathbf{C}_1\mathbf{D}_{1B})]\mathbf{D}_{2B}$ with his private keys $\mathbf{D}_{1B}$ and $\mathbf{D}_{2B}$ which would result in the plaintext $\mathbf{M}$. The usage of private ephemeral session key by Along is to ensure that the known plaintext attack will not be successful.

4 Conclusion

The AA$_\beta$-cryptosystem is secure as long as the LDEDLP is intractable. The LDEDLP has a preferred solution originating from a set of infinitely many solutions. The private key length still remains to be seen. The simplicity of the scheme can be gauged in terms of speed and computational effort to operate. It is known that RSA and ECC is of order $O(n^3)$ when encrypting a message block of length n whilst AA$_\beta$-cryptosystem is of order $O(n^2)$ (utilizes basic arithmetic operation of multiplication). The AA$_\beta$-cryptosystem also has the ability to encrypt large plaintext blocks without having to endure the computational intensive operations of the ECC, makes it a potential candidate that is “cryptographically efficient”.

References

[1] Ariffin,M.R.K., Abu,N.A.: AA$_\beta$-cryptosystem: A chaos based public key cryptosystem. Int. Jour. Cryptology Research. **1(2)**(2009) 149–163

[2] Ariffin,M.R.K., Abu,N.A.: Strengthening the AA$_\beta$-cryptosystem. Proc. Second International Cryptology Conference. (2010) 16–26

[3] Blackburn,S.R.: The Discrete Log Problem Modulo 1: Cryptanalyzing the Ariffin - Abu cryptosystem. J. Mathematical Cryptology. **4**(2010) 193–198

[4] Cohen,A.E., Parhi,K.K.: Implementation of Scalable Elliptic Curve Cryptosystem Crypto-Accelerators for GF(2^m). Conference Record of the Thirty-Eighth Asilomar Conference on Signals, Systems and Computers 1. (2004) 471–477

[5] Hermans,J. et.al.: Speed Records for NTRU. CT-RSA 2010, LNCS 5985. (2010) 73–88

[6] Hoffstein,J., Pipher,J., Silverman,J.H.: NTRU: A Ring Based Public Key Cryptosystem. Algorithmic Number Theory (ANTS III) Lecture Notes in Computer Science 1423. (1998) 267–288

[7] Scott,M.: When RSA is better than ECC, [Online]. Available: http://www.derkeiler.com/Newsgroups/sci.crypt/2008-11/msg00276.html (2008)

[8] Vanstone,S.: ECC holds key to next generation cryptography, [Online]. Available: http://www.design-reuse.com/articles/7409/ecc-hold-key-to-next-gen-cryptography.html (2006)

$\varGamma$-MAC$[H, P]$ - A new universal MAC scheme

Ewan Fleischmann, Christian Forler, and Stefan Lucks

Bauhaus-University Weimar, Germany. E-Mail:
`(Ewan.Fleischmann|Christian.Forler|Stefan.Lucks)@uni-weimar.de`

Abstract. In this paper we introduce $\varGamma$-MAC$[H, P]$, a new MAC scheme based on universal hash functions. An issue of Wegman-Carter-Shoup (WCS) based MACs, like the CWC-MAC [11] or GMAC [14], is that their security breaks apart in the nonce-reuse szenario [2,10]. Our new MAC scheme does not require a state or a (never-repeating) nonce for its security.
Most of the Block-Cipher based MACs are stateless. But, beside OMAC, they usually require at least two n-bit keys. For $\varGamma$-MAC$[H, P]$, only one single n-bit key suffices. Furthermore, a naive $\varGamma$-MAC$[H, P]$ instantiation based on $l+1$ block cipher invokations are needed to produce a security tag.
Depending on the length of the authenticated message, our implementation of $\varGamma$-MAC[G,AES-128] is faster than most other stateless MACs.

Keywords: universal hash, provable security, MAC.

1 Introduction

Message Authentication Code. Message authentication codes (MACs) are widely used cryptographic primitives – utilized to verify the integrity and authenticity of messages, assuming a secret key k shared by sender and receiver. MACs consist of two functions: MAC_k and VF_k. The authentication function MAC_k is used by the sender to generate a security tag $t = \mathrm{MAC}_k(m)$ for a message m. Given the pair (m, t), the receiver calls the verification function $\mathrm{VF}_k(m, t)$, which returns `true` if t actually has been generated as $t = \mathrm{MAC}_k(m)$. The security tag t ought to be short (typically 32–256 bit) to minimize the overhead for authentication. The attacker attempts to forge a message, i.e., to find new (m', t')-pairs with $\mathrm{VF}_k(m', t')$ =`true`. A MAC is secure if it is hard for the adversary to succeed. We consider chosen-plaintext existential forgery attacks, where the adversary is allowed to freely choose messages and nonces, and succeeds by forging the tag for any fresh message.

Universal hashing. Information-theoretically secure MACs have first been studied by Gilbert, Mac Williams and Sloane [8], and later by Wegman and Carter [18]. A strictly information-theoretical approach would require very long keys or greatly limit the number of messages to be authenticated under a given key. Thus, one typically combines the information-theoretical part – a universal hash function – with an additional function, which is modelled as a random function or permutation. There are two families of such MACs, which were studied so far.

Universal MAC schemes. The first family is due to Wegman, Carter and Shoup. We denote it as the "Wegman-Cater-Shoup" [16] (in short: "WCS$[H, F]$") approach. The WCS$[H, F]$-MAC is based on a family of 'ϵ almost XOR universal hash functions ("$\epsilon-AXU$") H, and a family of (pseudo-)random functions (PRF) F. For $h \in H$ and $f \in F$ the WCS-MAC is defined as $\mathrm{WCS}_{f,h}(m, z) = h(m) \oplus f(z)$, where m is the message and z is the nonce. The WCS approach allows it to use the $\epsilon - AXU$ hash function h several times, if and only if z is an never reused. If z is ever reused, one can compute the XOR $h(m) \oplus h(m')$ of the hashes of two different messages and typically break the MAC [10]. MACs following this approach are well studied and improved over the years for cryptographic purposes by Brassard [5], Krawczyk [12], Rogaway [15], Stinson [17], and other authors [1,3,4,7,9,14].

The second family follows the UMAC$[H, F]$ resp. FCH paradigm (we read this as "Function, Concatenation, Hash") from Black et al. [3]. We call MACs that fit into this scheme WMAC$[H, F]$, like Blake and Cochran in [2]. Let H a family of ϵ always universal ("$\epsilon - AU$") hash functions and F a family of PRFs. For For $h \in H$ and $f \in F$ the WMAC is defined as $\mathrm{WMAC}_{f,h}(m, z) := f(h(m), z)$ for a message m and a nonce z. Alike, as WCS$[H, F]$, the message is hashed using a randomly chosen hash function h out of an family of universal hash functions. In contrast to WCS, the hash output is not XOR-ed with the output of a random function, but is used as a part of the random function's input, jointly with the nonce z. Since the internal hash values $h(m)$ are only used as the input for a random function, the security of WMAC$[H, F]$ remains intact even if the nonce z is re-used. In fact, one doesn't actually need a nonce, but can securely use $t = f(h(m))$. While the nonce z may not be necessary to defend FCH-MACS against forgeries, many security protocols employ a time stamp, a sequence counter or something similar anyway. Using this auxiliary information as an additional "nonce" input to the authentication and to the verification function can be useful as a defense against "replay attacks" – i.e., resending an old message (the adversary hopes that it will be misunderstood in the new context) and the old authentication tag.

Our Contribution. At first we introduce ϵ-APU, a new class of universal hash functions. Informal we say H is ϵ-APU, if for a random chosen $h \in H$, it is very unlikely that 1) $h(m) = c$ for any tuple (m,c), and 2) $h(m) = h(m')$ for two distinct inputs m and m'. Then we present Γ-MAC$[H, P]$, a stateless MAC scheme based on a family of ϵ-APU has functions H and a (pseudo-)random permutation (PRP) $p \in P$. For $h \in H$, Γ-MAC$[H, P]$ is defined as $\Gamma_{h(m)} := p_{h(m)}(|m|)$. Unlike WCS$[H, F]$ and WMAC$[H, F]$, our MAC scheme needs only one key to determine unambiguously the universal hash function h from H. The PRP p is determine unambiguously from the output value of h. Γ-MAC$[H, P]$ is the second stateless universal MAC scheme – beside WMAC$[H, F]$– that is known in literature. The major advantage of Γ-MAC$[H, P]$ over WMAC$[H, F]$ is that we only need *one* key. On closer look, this can be also a disadvantage because block cipher must be resistant against related key attacks. Therefore, we present a ϵ-AXPU hash families. Instantiated with such a hash family the used block cipher must not be resistant to related key attacks that exploit a XOR difference between keys. Furthermore, the key scheduler must be invoked if the authentication or verification function is called. A analysis showed that the performance of Γ-AES, a Γ-MAC$[H, P]$ instance based on AES-128, does not suffer from this key scheduler invocation issue.

Related Work. Modern universal hash function based MACs, like GMAC from McGrew and Viega GMAC [14], VMAC from Krovetz and Dai [6, 13] or Bernstein's Poly1305-AES [1] are similar in spirit to Γ-MAC$[H, P]$, but employ two n-bit keys, while a single one suffices for Γ-MAC$[H, P]$. Like GMAC, the presented instance of Γ-MAC$[H, P]$ employs universal hashing based on Galois field multiplications, which run very efficiently in hardware. Handschuh and Preneel [10] pointed out that MACs based on universal hashing are brittle, with respect to their combinatorial properties, and that some are extremely vulnerable to nonce reuse attacks. Black and Cochran [2] recently presented WMAC$[H, F]$. This MAC scheme matches the security bounds given by the best known attacks from Handschuh and Preneel.

References

[1] Daniel J. Bernstein. The Poly1305-AES Message-Authentication Code. In *FSE*, pages 32–49, 2005.

[2] John Black and Martin Cochran. MAC Reforgeability. In *FSE*, pages 345–362, 2009.

[3] John Black, Shai Halevi, Hugo Krawczyk, Ted Krovetz, and Phillip Rogaway. UMAC: Fast and Secure Message Authentication. In *CRYPTO99*, pages 216–233, 1999.

[4] Martin Boesgaard, Ove Scavenius, Thomas Pedersen, Thomas Christensen, and Erik Zenner. Badger - A Fast and Provably Secure MAC. Cryptology ePrint Archive, Report 2004/319, 2004. http://eprint.iacr.org/.

[5] Gilles Brassard. On Computationally Secure Authentication Tags Requiring Short Secret Shared Keys. In *CRYPTO*, pages 79–86, 1982.

[6] Wei Dai and Ted Krovetz. VHASH Security. Cryptology ePrint Archive, Report 2007/338, 2007. http://eprint.iacr.org/.

[7] Mark Etzel, Sarvar Patel, and Zulfikar Ramzan. SQUARE HASH: Fast Message Authenication via Optimized Universal Hash Functions. In *CRYPTO*, pages 234–251, 1999.

[8] E. N. Gilbert, F. J. MacWilliams, and N. J. A. Sloane. Codes which detect deception. *Bell System Tech. J.*, 53:405–424, 1974.

[9] Shai Halevi and Hugo Krawczyk. MMH: Software Message Authentication in the Gbit/Second Rates. In *FSE*, pages 172–189, 1997.

[10] Helena Handschuh and Bart Preneel. Key-Recovery Attacks on Universal Hash Function Based MAC Algorithms. In *CRYPTO 2008: Proceedings of the 28th Annual conference on Cryptology*, pages 144–161, Berlin, Heidelberg, 2008. Springer-Verlag.

[11] Tadayoshi Kohno, John Viega, and Doug Whiting. CWC: A High-Performance Conventional Authenticated Encryption Mode. In *FSE*, pages 408–426, 2004.

[12] Hugo Krawczyk. LFSR-based Hashing and Authentication. In *CRYPTO*, pages 129–139, 1994.

[13] Ted Krovetz. Message Authentication on 64-Bit Architectures. In *Selected Areas in Cryptography*, pages 327–341, 2006.

[14] David A. McGrew and John Viega. The Security and Performance of the Galois/Counter Mode (GCM) of Operation. In *In INDOCRYPT, volume 3348 of LNCS*, pages 343–355. Springer, 2004.

[15] Phillip Rogaway. Bucket Hashing and its Application to Fast Message Authentication. In *CRYPTO*, pages 29–42, 1995.

[16] Victor Shoup. On Fast and Provably Secure Message Authentication Based on Universal Hashing. In *In Advances in Cryptology CRYPTO 96*, pages 313–328. Springer-Verlag, 1996.

[17] Douglas R. Stinson. Universal Hashing and Authentication Codes. *Des. Codes Cryptography*, 4(4):369–380, 1994.

[18] Mark N. Wegman and J. Lawrence Carter. New hash functions and their use in authentication and set equality. *JCSSBM*, 22(3):265–279, June 1981.

A new secure sketch based on the discrete logarithm problem

Max Grosse and Stefan Lucks

Bauhaus–Universität Weimar
{max.grosse,stefan.lucks}@uni-weimar.de

Abstract. The concept of secure sketches has been previously presented as a special form of *fuzzy* commitments. Contrary to common commitment schemes, fuzzy commitments allow for unlocking the committed value also using a witness sufficiently *close* to the exact one. Previous approaches towards secure sketches, however, rely on error correcting codes and provide an information theoretic security. We present a new approach that builds upon computational security by making use of the discrete logarithm problem. We show that it is advantageous in several aspects and describe a practical application in the field of image processing.

Keywords: secure sketch, discrete logarithm, fuzzy commitment scheme, image processing

1 Introduction

Matching of two different datasets can often be achieved by comparison of the associated cryptographic hash values. However, due to the nature of cryptographic hash functions, minutest changes to one of the datasets results in tremendous changes in the hash value. Consequently, hash functions are not well suited for matching of *similar*, but not equal, datasets.

A similar problem arises when using commitment schemes. In such a scheme, a value is committed into a commitment that can be shared, while the value itself remains secret. Revealing a witness at a later time allows for decommitting the commitment and so revealing the original value, additionally proving the authenticity as the commitment had been published. In this case, only the exact witness can be used for decommitting, although there might be situations in which the witness is degraded to some extent.

As an example, consider a court proceeding where a photograph showing several people is used as evidence. In order to protect the identity of eyewitnesses on the picture, the photograph should not be published. However, at any time it should be possible to prove the existence of the evidence, and more importantly, the information provided by it. One possibility would be to publish only a cryptographic hash value of the image. A printed copy containing only parts of the evidence and brought to court cannot be validated then. The printed copy is partly degenerated by the printing process, therefore the hash value is likely to no longer match. Direct comparison, however, is not desirable either, as this would require revealing the original. To solve this problem, we describe the use of a fuzzy *secure sketch*, where any degenerated partial copy of a secret commitment can be validated without revealing the original at all.

2 Related work

This work builds on the previous approaches towards fuzzy commitments and secure sketches as presented by Juels and Wattenberg [1], as well as Dodis et al. [2]. Instantiations of these approaches make use of error correcting codes and provide an information theoretic security. Nevertheless, several problems arise in these approaches, affecting both the *blinding* as well as *binding* properties of the commitment schemes involved, resulting in significant difficulties at practical use.

For linear error correcting codes, the *code-offset construction* is defined for a value w to be committed, $F(w)$ the commitment and h a hash function as $F(w) = (h(w), \mathsf{syn}(w))$, with $\mathsf{syn}(w)$ the error correction syndrome.

Dodis et al. present [2] a rigorous formalization of secure sketches, where a secure sketch basically consists of two functions, SS and Rec, so that for a value w to be committed $\mathsf{Rec}(w', \mathsf{SS}(w)) = w$ for any value w' sufficiently close to w. In case of the code-offset construction, $\mathsf{SS}(w)$ can be defined as $\mathsf{SS}(w) = \mathsf{syn}(w)$, while Rec provides error correction to recover w from w' and the syndrome of w.

3 Problems of previous approaches based on error correcting codes

Several problems may arise from the use of the previously described schemes, especially through the use of a $[n, k, 2t+1]-$error correcting code, where k bits are encoded into n-bit code words with t-bit errors correctable. These problems are as follows.

2nd Preimages. The additional storage of a hash value along the sketch is crucial, as 2nd preimages of a sketch alone are potentially trivial. The parity check matrix $\mathbf{P}$ has to be known for a linear error correcting code, as $\mathsf{syn}(w) = w\mathbf{P}^T$. However, because $\mathbf{P}^T \cdot \mathbf{P} = \mathbf{1}$, a preimage w' for which $\mathsf{syn}(w') = \mathsf{syn}(w)$ can be simply found, as $\mathsf{syn}(w)\mathbf{P} = w'$.

All Preimages. In a similar fashion, it is also possible to enumerate all preimages, that are all w' for which $\mathsf{syn}(w') = \mathsf{syn}(w)$. There are at most 2^{2k-n} distinct w' for which $\mathsf{syn}(w') = \mathsf{syn}(w)$. Having found one preimage (as described before), so that $s = w'\mathbf{P}^T$, all other preimages can be determined by finding the kernel of $\mathbf{P}^T$. Because $\mathbf{P}^T$ has to be known, by using basic linear algebra it is trivial to compute its kernel. Once the kernel has been computed, all 2^{2k-n} preimages can be enumerated in $O(2^{2k-n})$, while precomputing the kernel can be done in negligible time and space.

For secure sketches based on linear block codes, it is imperative to store the hash value of the secret as well, as described by the original authors, because preimages are trivial to be found.

However, even if the hash value is stored as well, there are exactly 2^{2k-n} possible preimages that can be simply enumerated. Therefore, an adversary is required to perform 2^{2k-n} evaluations of the hash function to find the correct preimage. For every $[n, k, 2t+1]$-code, the ratio of k and n determines the capability of correcting errors, with a larger k/n resulting in more errors correctable, but also fewer possibilities for an adversary to test. This introduces a significant problem, as a wrong choice of an error correcting codes can result in attacks becoming trivial, while a conservative choice of a code may provide only very limited error correcting capabilities.

4 Secure sketch based on the DLP

A computationally secure sketch can be built on the discrete logarithm problem (DLP). If the discrete logarithm problem is *hard*, the associated secure sketch problem can be considered as hard as well. For a group G with a generator g of order n, the discrete logarithm problem is solving for x in $y = g^x$, with only y, g known. With q the largest prime factor of n, any generic algorithm is bounded by $\Omega(\sqrt{q})$ group operations [3]. We now define Rec and SS to build a secure sketch, which draws its security from the difficulty of computing the discrete logarithm as follows. For a value w to be committed,

$$s = \mathsf{SS}(w) = g^{-w}. \tag{1}$$

Now assume there is a function Λ solving the discrete logarithm problem, i.e. $\Lambda(g^x) = x$. Then we can define Rec as

$$\mathsf{Rec}(w', s) = w' - \Lambda(g^{w'} \cdot s) = w' - \Lambda(g^{w'} \cdot g^{-w}) = w, \tag{2}$$

as $w' - \Lambda(g^{w'} \cdot g^{-w}) = w' - \Lambda(g^{w'-w}) = w' - (w' - w) = w$. Even so, there is no generic function Λ known to efficiently compute the discrete logarithm. In fact, if such a generic Λ existed, any arbitrary w' could be used, resulting in an absolutely non-secure sketch.

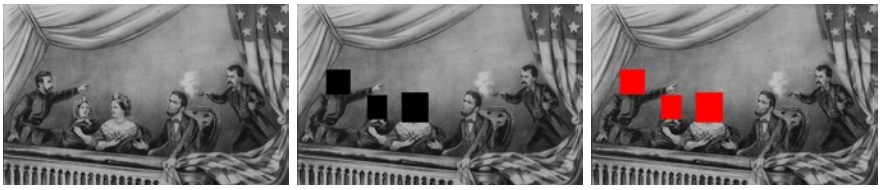
Thus, we define Λ to be limited to only a small subset of exponents,

$$\Lambda(g^{w'} \cdot g^{-w}) = w' - w \text{ iff } |w' - w| \leq t, \tag{3}$$

for t describing the error correction capabilities of the secure sketch, and $t \ll q$. As a matter of fact, such a function Λ then can be easily provided and implemented. One possibility would be a simple table–lookup of all t possible values. Generally, any efficient algorithm for computing the discrete algorithm inside a given range $[0, t]$ suffices, for instance Pollard's Kangaroo [4] or a slightly modified baby–step giant–step algorithm [5].

The presented secure sketch is in fact secure, as it can be shown that in the general case, finding a w' so that $\mathsf{Rec}(w', \mathsf{SS}(w)) = w$ without knowing w is lower bounded by $O(\sqrt{q/t})$, with q the largest prime factor of the order of the group used.

5 Results



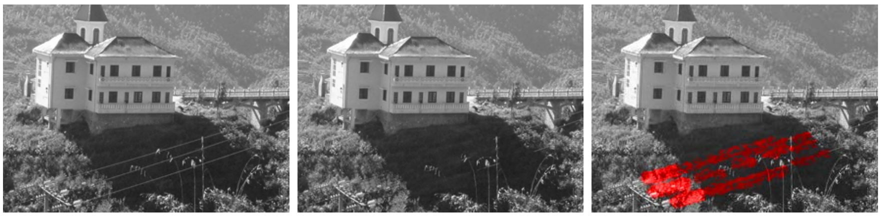
(a) Original (b) Image used to unlock (c) Recovered

Fig. 1. The original image (a) is encoded inside a secure sketch. A censored version (b) is provided to unlock the sketch. The original image can be verified to be authentic, beside the regions censored (c).

The initial example of the court proceeding has been implemented using the proposed secure sketch based on the discrete logarithm problem. For this, several image pixels are concatenated into a block and the secure sketch is computed and stored for each such block. A modified image, for example one where the faces of eyewitnesses have been censored, can be verified against the encoded sketch. Regions in which the validation fails because of the censored parts can be highlighted. This is illustrated in figure 1. In the same fashion, if a tampered image is provided for validation against a sketch, modified regions are clearly exposed. This is illustrated in figure 2, where an image has been carefully manipulated to no longer show power lines. Validation against the original immediately highlights those regions, without the need for revealing the original. Note that the exact shape of the power lines remains secret as well, only the presence of modifications in these regions is indicated. All these examples are robust against a limited amount of noise, possibly due to image compression. Furthermore, using a more sophisticated approach where also certain image features are stored along the sketch, it is even possible to successfully validate slightly distorted copies, without significantly compromising the security.

6 Summary

Previous approaches towards secure sketches exhibit several problems arising through the use of error correcting codes. We present a new concept of secure sketches based on computational security instead of information theoretical security as previous approaches do. Specifically, the discrete logarithm problem



(a) Original (b) Tampered (c) Reconstruction

Fig. 2. The original image (a) is encoded inside a secure sketch. When tampered version (b) of the original, where the power lines have been removed, is used to unlock the sketch, the differences can be visualized (c) so any modifications become apparent. However, the authenticity of the remainder is proven.

has been facilitated to provide a novel secure sketch, which draws its hardness from the hardness of the discrete logarithm problem. A practical implementation has been shown, where images are encoded into a secure sketch such that degraded copies can be validated without the need to reveal the original at all.

References

1. Juels, A., Wattenberg M.: A fuzzy commitment scheme. In: Proceedings of the 6th ACM conference on Computer and communications security, pages 2836. ACM (1999)
2. Dodis, Y., Reyzin, L., Smith, A.: Fuzzy extractors: How to generate strong keys from biometrics and other noisy data. In: Advances in cryptology-Eurocrypt 2004, pages 523540. Springer (2004)
3. Shoup, V.: Lower bounds for discrete logarithms and related problems. In: Proceedings of the 16th annual international conference on Theory and application of cryptographic techniques, pages 256266. Springer-Verlag (1997)
4. Pollard, J.M.: Monte Carlo methods for index computation (mod p). Mathematics of computation, 32(143):918924 (1978)
5. Shanks, D.: Class number, a theory of factorization, and genera. In: Proceedings of the Symposium Pure Mathematics, volume 20, pages 415440. American Mathematical Society (1971)

A Lightweight Pseudorandom Number Generator For EPC Class 1 Gen2 RFID Tags

Kalikinkar Mandal, Xinxin Fan, and Guang Gong

Department of Electrical and Computer Engineering
University of Waterloo
Waterloo, Ontario, N2L 3G1, CANADA
{kmandal, x5fan, ggong}.uwaterloo.ca

1 Introduction

Radio Frequency Identification (RFID) is a promising technology for automatic identification of remote objects. For most RFID applications, security is an important or even crucial requirement. Since most protocols for securing RFID systems proposed so far are based on the use of an on-board true random and/or pseudorandom number generator (TRNG/PRNG), a number of solutions have been proposed in literature for implementing TRNGs/PRNGs on RFID tags [1, 2, 6, 8, 10]. In particular, the EPCglobal Class-1 Generation-2 (EPC C1 Gen2 in brief) standard [3] uses random numbers in the tag identification protocol. All of the proposals for TRNGs are based on analog circuits that sample a random physical phenomenon like thermal noise. To the best of our knowledge, only three PRNGs have been proposed for EPC C1 Gen2 tags [2, 8, 10], among which two proposals use TRNGs as a component and the security properties of these two PRNGs rely on the security of TRNGs. Considering the high power consumption, large area and low throughput of TRNGs, we propose a lightweight PRNG for low-cost EPC C1 Gen2 tags in this contribution. The basic idea of our design is to replace the TRNG in [2, 8] by a lightweight pseudorandom sequence generator with good statistical properties. To this end, nonlinear feedback shift registers (NFLSRs) have been fully exploited in our design. An estimation of the hardware complexity shows that the proposed PRNG can be implemented using around $1,242$ logic gates.

2 Description of the Proposed PRNG

The proposed PRNG is composed of two main building blocks. The first one consists of two NLFSRs of length 17 and 18, each one generating a span-n-sequence with the optimal linear complexity, whereas the second one includes a nonlinear feedback shift register and a WG transformation [4] module. In our design, the binary sequence generated by the first building block is converted to the sequence over the finite field $\mathbb{F}_{2^5}$ and this sequence is used to select a characteristic polynomial for the recurrence relation in the second building block. The final output sequence is filtered by the WG transformation and n-bit random numbers are generated by taking disjoint n-bit sequences from the final output sequence. An overview of the proposed PRNG is illustrated in Fig. 1 and a more detailed description is presented in the following subsections.

2.1 Building Block I: An Alternative to TRNG

The first building block contains two NLFSRs whose lengths (i.e., 17 and 18) are chosen to be coprime in order to achieve the maximum period. The reason that the two smaller length NLFSRs are used instead of a longer one is because it is impossible to generate shift distinct sequences from a longer length NLFSR for different initial states. In our design, the WG transformation over $\mathbb{F}_{2^5}$ is used as a nonlinear feedback function to generate span-n-sequences. For $m = 5$, the WG permutation is

$$WGP_5(x) = x + (x+1)^5 + (x+1)^{13} + (x+1)^{19} + (x+1)^{21}, x \in \mathbb{F}_{2^5},$$

and the WG transformation over $\mathbb{F}_{2^5}$ is given by

$$f(x) = \mathrm{Tr}(WGP_5(x)) = \mathrm{Tr}(x^{19}),$$

where $\mathrm{Tr}(\cdot) : \mathbb{F}_{2^5} \mapsto \mathbb{F}_2$ is a trace function over $\mathbb{F}_{2^5}$. The n-stage nonlinear recurrence relation is defined as

$$s_{n+k} = s_k + f(x^d),\ x = (s_{r_1+k}, s_{r_2+k}, \ldots, s_{r_5+k}) \in \mathbb{F}_{2^5} \text{ and } s_i \in \mathbb{F}_2$$

for all $k \geq 0$, and $0 < r_1 < r_2 < \ldots < r_5 < n$ are tap positions of two NLFSRs. Using the parameters and recurrence relations in Table 1, we can generate two span-n-sequences $\mathbf{b} = \{b_i\}_{i\geq 0}$ and $\mathbf{c} = \{c_i\}_{i\geq 0}$ with NLFSR1 and NLFSR2, respectively. The output sequence of the first building block is denoted by $\mathbf{s} = \{s_i \mid s_i = b_i \oplus c_i, i \geq 0\}$, which is almost balanced and has the following statistical properties: a) The period is $(2^{18} - 1)(2^{17} - 1) \approx 2^{35}$; b) The imbalance range is 4; and c) The linear span is $2^{17} - 2 + 2^{18} - 2 \approx 2^{18.585}$.

Table 1. Parameters and Statistical Properties of Two NLFSRs

NLFSR	Length	Decimation	Primitive polynomial	Tap positions	Period	Linear
	n	d	$p(x)$ to generate $\mathbb{F}_{2^5}$	$(r_1, r_2, r_3, r_4, r_5)$		Span
NLFSR1	18	3	$1 + x + x^3 + x^4 + x^5$	4, 7, 8, 10, 15	$2^{18} - 1$	$2^{18} - 2$
NLFSR2	17	3	$1 + x + x^3 + x^4 + x^5$	4, 7, 8, 9, 12	$2^{17} - 1$	$2^{17} - 2$

We now generate a sequence $\mathbf{t}= \{t_k\}_{k\geq 0}$ over $\mathbb{F}_{2^5}$ from $\mathbf{s}$ as follows

$$t_k = (s_{5k}, s_{5k+1}, s_{5k+2}, s_{5k+3}, s_{5k+4}) \in \mathbb{F}_{2^5}, \forall k \geq 0.$$

This sequence is used to select a characteristic polynomial for the second building block.

2.2 Building Block II: Pseudorandom Number Generator

The second building block consists of a NLFSR and a WG transformation module to filter the sequence over the field $\mathbb{F}_{2^5}$. Let the length of the NLFSR3 be $l = 6$ and the primitive polynomial be $g(x) = x^6 + x + \gamma$, where $\gamma = \alpha^{15} \in \mathbb{F}_{2^5}$. The recurrence relation[1] is defined as

$$a_{k+6} = \gamma a_k + a_{k+1} + WGP_5(a_{k+5}) + t_k, a_i \in \mathbb{F}_{2^5}, \tag{1}$$

where $\mathbf{t} = \{t_k\}_{k\geq 0}$ is the sequence over $\mathbb{F}_{2^5}$ that is defined in the previous subsection. Note that the period of the sequence $\mathbf{a} = \{a_k\}_{k\geq 0}$ is at least that of $\mathbf{t}$. Moreover, the final output sequence of the second building block is defined by $o_k = f(a_{5+k})$, for $k \geq 0$.

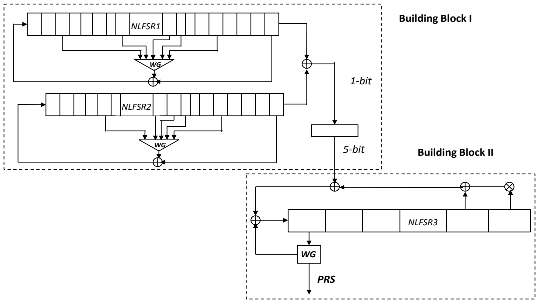


Fig. 1. Diagram of the PRNG for EPC C1 Gen2 Tags

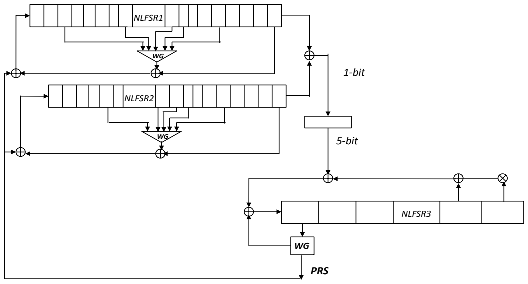


Fig. 2. The Key Initialization Procedure

It can be proved that the recurrence relation in Eq. (1) has a *multiple-polynomial* LFSR form with the characteristic polynomial $q_k(x) = \gamma + x + \phi(r_k)x^5 + x^6, r_k \in \mathbb{F}_{2^5}$. Moreover, one out of 2^5

[1] The recurrence relation excluding t_k is defined in [9, 7] for key initialization of the WG cipher.

characteristic polynomials is chosen at each clock cycle and the choice of the polynomial $q_k(x)$ depends on the sequence $\mathbf{t}$. In addition, the polynomial $g(x)$ is chosen such that among 32 characteristic polynomials seven of them (i.e., the maximum number) are primitive. In [8], one of the eight primitive polynomials is chosen by a decoding logic at each clock cycle. However, in our case the number of characteristic polynomials is much more than those in [8] and the recurrence relation in Eq. (1) is used to select a characteristic polynomial instead of a decoding logic.

2.3 System Initialization

The proposed PRNG has an internal state of 65 bits, including a 45-bit secret seed k as well as a 20-bit initial vector (IV). While the secret seed and the IV are preloaded into RFID tags at the very beginning, the 20-bit IV is also updated at the end of each protocol session. Before generating random numbers, a 36 rounds of initialization phase is applied to mix the key and IV properly. In our design, the secret seed and IV are preloaded as follows: the first consecutive 11, 12 and 22 positions of the NLFSR1, NLFSR2 and NLFSR3 are respectively reserved for key bits, whereas the remaining positions in each NLFSR are for the IV. The initialization process is illustrated in Fig. 2. During the initialization phase the internal states of the three NLFSRs are updated as follows:
$b_{k+18} = b_k + f(x^3) + o_k, x = (b_{k+4}, b_{k+7}, b_{k+8}, b_{k+10}, b_{k+15}),\ k \geq 0,\ o_k = 0$ for $k = 0$,
$c_{k+17} = c_k + f(y^3) + o_k, y = (c_{k+4}, c_{k+7}, c_{k+8}, c_{k+9}, c_{k+12}),\ k \geq 0,\ o_k = 0$ for $k = 0$,
$s_k = b_k + c_k,\ k \geq 0$,
$t_k = (s_k, s_{k+1}, s_{k+2}, s_{k+3}, s_{k+4}),\ k \geq 0$,
$a_{k+6} = a_k + a_{k+1} + WGP_5(a_{k+5}) + t_k,\ k \geq 0.$

3 Security Properties

We analyzed the security properties of the proposed PRNG by performing several cryptographic statistical tests on several sets of pseudorandom sequences generated by our PRNG for different initial states. We performed all the statistical tests that are proposed in the EPC C1 Gen2 standard [3] as well as in the NIST standard [12], respectively.

According to the EPC C1 Gen2 standard [3], a true random or pseudorandom number generator must satisfy the following three statistical properties:

- **Probability of a single sequence:** The probability that any 16-bit random sequence ($RN16$) drawn from the PRNG has value j, shall be bounded by $\frac{0.8}{2^{16}} < \Pr(RN16 = j) < \frac{1.25}{2^{16}}$, for any j.
- **Probability of simultaneously identical sequences:** For a tag population up to ten thousand tags, the probability that any of two or more tags simultaneously generate the same sequence of bits shall be less than 0.1%, regardless of when the tags are energized.
- **Probability of predicting a sequence:** A given sequence drawn from the PRNG 10ms after the end of transmission shall not be predictable with a probability grater than 0.025% if the outcomes of prior draws from PRNG, performed under identical conditions, are known.

We implemented our PRNG in software for checking whether the proposed PRNG meets the above three criteria. To verify the first criterion, we generated 18 different test sequences for different initial states of the NLFSRs and we calculated the probability of occurrence of 16-bit values. Our experimental results show that $\Pr(RN16 = j)$ lies between $\frac{0.9628}{2^{16}}$ and $\frac{1.0428}{2^{16}}$, which are better bounds than those obtained in [8]. With respect to the second criterion, our PRNG can generate up to $2^{45} - 1$ shift distinct sequences for different keys to each tag. Thus the probability that any two tags will generate the same sequence is 2^{-45} that is much less than 0.1%. For the third criterion, given a 16-bit random number, an attacker can recover the internal state of the NLFSR3 with probability 2^{-24} and get 80 bits of the sequence $\mathbf{s}$. To obtain the next 16-bit random number from the given one, the adversary needs to know the next consecutive 80 bits of the sequence $\mathbf{s}$ and the internal state of the NLFSR3. The 80 bits can be obtained either by guessing or obtaining about $\frac{2^{18.58}}{5} = 2^{16.26}$ consecutive random numbers. Due to the high linear span of the sequence $\mathbf{s}$, it is impossible to generate the next consecutive 80 bits from previous known 80 bits in practice. Furthermore, it is also difficult for an adversary to intercept $2^{16.28}$ consecutive random numbers in one protocol session because the communication session in RFID systems is usually quite short and the IV is different and the secret seed can also be updated for different sessions. Hence, the attacker can guess the next 16-bit random number with the better probability 2^{-16}, which is much less than 0.025% as specified in the EPC C1 Gen2 standard.

To measure the linear dependency between each n-bit output and previous n-bit output, we performed a serial correlation test on the sequences generated by the proposed PRNG. We generated 18 distinct sequences for different initial values and calculated the serial correlation coefficient for 1-bit, 1-byte and 2-byte lag. Our experimental results demonstrate that the serial correlation coefficients are close to zero, which indicates the good pseudorandomness of the generated sequences.

Different from the statistical tests in the EPC C1 Gen2 standard, the NIST test suite contains 15 demanding statistical tests for characterizing the randomness of a binary sequence. According to the NIST specification [12], a PRNG passes the test suite successfully if it passes all the tests simultaneously with a proportion of 96%. In our experiment, 10 test sequence (TS) sets are generated, each of which has 100 different sequences with different initial values and has a length of 2^{25}. We computed the proportion values for each TS set and listed the test results[3] for 5 out of 10 TS sets in Table 2. It is not difficult to find out that each TS set can pass the NIST test suite successfully.

Table 2. NIST Test Suite results of our proposal

Tests	TS1	TS2	TS3	TS4	TS5
	proportion	proportion	proportion	proportion	proportion
Frequency	1.00	0.98	1.00	0.99	1.00
Block-frequency	1.00	0.99	0.98	1.00	1.00
Cumulative-sum	1.00, 1.00	0.98, 0.99	1.00, 0.99	0.99, 0.99	1.00, 1.00
Runs	1.00	0.99	0.98	0.99	1.00
Longest-run	0.99	0.98	0.98	0.99	0.99
Rank	0.99	0.98	0.99	0.99	0.99
FFT	1.00	1.00	0.98	1.00	1.00
Overlapping-templates	0.99	0.99	0.98	0.97	0.99
Universal	0.99	0.98	0.98	0.99	0.99
Approx. entropy	0.98	1.00	0.98	0.99	0.98
Serial	0.99, 0.99	0.99, 1.00	1.00, 0.99	0.97, 0.98	0.99, 0.98
Linear-complexity	0.98	0.99	1.00	0.97	0.98
Random-excursions	0.99, 0.99	0.99, 0.99	0.98, 1.00	1.00, 0.99	0.99, 0.99
	1.00, 0.97	0.97, 0.99	1.00, 1.00	0.99, 1.00	1.00, 0.97
	0.99, 1.00	1.00, 0.99	0.99, 0.99	0.98, 1.00	0.99, 1.00
	0.99, 0.99	0.97, 0.99	0.98, 1.00	0.99, 0.99	0.99, 0.99
Random-excur-variant	0.98, 0.99, 0.99	1.00, 1.00, 1.00	1.00, 1.00, 1.00	0.99, 0.99, 0.99	0.98, 0.99, 0.99
	1.00, 1.00, 0.99	1.00, 0.99, 0.99	1.00, 1.00, 1.00	1.00, 1.00, 1.00	1.00, 1.00, 0.99
	0.99, 0.99, 0.99	0.99, 0.99, 1.00	1.00, 1.00, 1.00	1.00, 0.99, 1.00	0.99, 0.99, 0.99
	1.00, 0.98, 1.00	1.00, 0.99, 0.99	0.99, 1.00, 1.00	0.99, 1.00, 0.99	1.00, 0.99, 1.00
	1.00, 0.97, 0.99	0.99, 1.00, 1.00	1.00, 1.00, 1.00	0.99, 0.99, 0.99	1.00, 0.97, 0.99
	0.99, 0.99, 0.99	1.00, 1.00, 1.00	1.00, 0.99, 0.99	1.00, 0.99, 0.99	0.99, 0.99, 0.99

4 Hardware Complexity

Besides the good randomness properties, the proposed lightweight PRNG can also meet the stringent requirements of the EPC C1 Gen2 standard regarding to the hardware complexity. A PRNG is expected to be implemented with a small number of logic gates according to the EPC C1 Gen2 standard [3] and a usual rule of thumb is that the security functionality in EPC tags costs between 2000 and 5000 logic gates [11]. A rough estimation in Table 3 shows that the proposed PRNG can be implemented in hardware with around $1,242$ logic gates, which perfectly matches the requirements of the EPC C1 Gen2 standard. Moreover, our PRNG has a lower hardware complexity than that in [10]. When compared to the PRNG proposed in [8], our design costs more logic gates in order to replace the TRNG in [8]. However, if we only compare the hardware implementation cost for the pseudorandom number generator module (i.e., the building block II in our design) in both proposals, our design only needs 127 less logic gates than that in [8].

[3] Non-overlapping template matching test results are not given in the table because of 148 entries.

Table 3. The Hardware Complexity of the Proposed PRNG

Component	Quantity	Function	Gate Count
LFSR18	1	Generation of a span-n-sequence $\mathbf{b} = \{b_i\}_{i \geq 0}$	216
LFSR17	1	Generation of a span-n-sequence $\mathbf{c} = \{c_i\}_{i \geq 0}$	204
LFSR5	1	5-bit storage of the sequence $\mathbf{t} = \{t_k\}_{k \geq 0}$	60
LFSR5×6	1	Generation of a span-n-sequence $\mathbf{a} = \{a_k\}_{k \geq 0}$	360
WG	1	WG transformation	133
XOR1	13	1-bit exclusive-OR operation	33
Multiplier5	1	5-bit multiplication over $\mathbb{F}_{2^5}$	29
Control (20%)		—	207
Total		—	$1,242$

With respect to the time delay for generating the first 16-bit pseudorandom number, our design totally requires 116 clock cycles, including 36 clock cycles for the initialization and $5 \times 16 = 80$ clock cycles for the generation of the first 16-bit random number. After that, each 16-bit random number can be obtained every 80 clock cycles. Assuming that the EPC tags run at the clock frequency of 100 KHz and two 16-bit random numbers are needed for the tag identification protocol according to the EPC C1 Gen2 standard, one can identify about 510 tags in one second by using our PRNG.

5 Conclusions

In this paper we propose a lightweight pseudorandom number generator in compliance to EPC Class-1 Generation-2 standard. Considering the high power-consumption, large area and low throughput of TRNGs, we replace the TRNG used in previous works by a PRNG with good statistical properties. In our design, the pseudorandom sequence is generated using a nonlinear feedback shift register in which the nonlinear recurrence relation can be treated as a *multiple-polynomial* LFSR form. Moreover, the statistical tests specified by the EPC C1 Gen2 and the NIST standards are employed to characterize the security properties of the proposed PRNG. In addition, a complexity estimation shows that the proposed PRNG can be implemented in hardware using around $1,242$ logic gates and can generate a 16-bit random number every 80 clock cycles after an initialization process of 36 clock cycles.

References

1. G.K. Balachandran, and R.E. Barnett. A 440-nA True Random Number Generator for Passive RFID Tags. *Circuits and Systems I: Regular Papers, IEEE Transactions on*, 55(11):3723 –3732, dec. 2008.
2. W. Che, H. Deng, X. Tan, and J. Wang. A Random Number Generator for Application in RFID Tags. In Cole, P.H. and Ranasinghe, D.C. (Eds.). In *Networked RFID Systems and Lightweight Cryptography*, Chapter 16, pages 279–287. Springer-Verlag, 2008.
3. EPCglobal(2008). EPC Radio-Frequency Identification Protocol Class-1 Generation-2 UHF RFID for Communication at 860-960 MHz, `http://www.epcglobalinc.org/`.
4. G. Gong, and A. Youssef. On welch-gong transformation sequence generators. In Douglas Stinson and Stafford Tavares, editors, *Selected Areas in Cryptography*, volume 2012 of *Lecture Notes in Computer Science*, pages 217–232. Springer Berlin / Heidelberg, 2001.
5. M. Hell, T. Johansson, and W. Meier. Grain: a stream cipher for constrained environments, `http://www.ecrypt.eu.org/stream/`.
6. D.E. Holcomb, W.P. Burleson, and K. Fu. Initial SRAM state as a fingerprint and source of true random numbers for RFID tags. In *In Proceedings of the Conference on RFID Security*, 2007.
7. C. Lam, M. Aagaard, and G. Gong. Hardware Implementations of Multi-output Welch-Gong Ciphers, CACR 2011-01, `http://www.cacr.math.uwaterloo.ca/`.
8. J. Melia-Segui, J. Garcia-Alfaro, and J. Herrera-Joancomarti. Analysis and Improvement of a Pseudorandom Number Generator for EPC Gen2 Tags. In *Proceedings of the 14th International conference on Financial Cryptograpy and Data Security*, FC'10, pages 34–46, Berlin, Heidelberg, 2010. Springer-Verlag.
9. Y. Nawaz, and G. Gong. WG: A family of stream ciphers with designed randomness properties. *Information Sciences*, 178(7):1903 – 1916, 2008.
10. P. Peris-Lopez, J. C. Hernandez-Castro, J. M. Estevez-Tapiador, and A. Ribagorda. LAMED - A PRNG for EPC Class-1 Generation-2 RFID specification. *Comput. Stand. Interfaces*, 31:88–97, January 2009.
11. D. C. Ranasinghe, and P. H. Cole. An Evaluation Framework. In *Networked RFID Systems and Lightweight Cryptography*, pages 157–167. Springer-Verlag, 2008.
12. A. Rukhin, J. Soto, J. Nechvatal, E. Barker, S. Leigh, M. Levenson, D. Banks, A. Heckert, J. Dray, S. Vo, A statistical test suite for random and pseudorandom number generators for cryptographic applications, `http://csrc.nist.gov/rng/`, Technical Report,2001.

Related-key attacks on the full GOST block cipher with 2 or 4 related keys

Marina Pudovkina and George Khoruzhenko

National Nuclear Research University
(Moscow Engineering Physics Institute)
Moscow, Kashirskoe shosse, 31

Abstract. The GOST 28147-89 block cipher is a Russian standard for encryption. We describe two attacks with 4 related keys and one attack using 2 related keys. The complexities of the attacks depend on properties of s-boxes. For some classes s-boxes the attacks are practical.

Keywords: related-key rectangle attack, truncated differential attack, differential attack, GOST block cipher

1 Introduction

The GOST block cipher is a Russian standard for encryption. It was standardized in 1989. All governmental organizations and some commercial organizations have information security systems based on this standard. In 2010 the GOST block cipher was submitted to ISO 18033, to become a worldwide industrial encryption standard. The block cipher GOST is based on the Feistel scheme. It has 32 rounds, 64-bit blocksize, and 256-bit keysize and iterates a round function f composed of a key addition modulo 2^{32}, eight bijective 4×4-bit s-boxes $s_i, 1 \leq i \leq 8$, and cyclic rotations by 11 bits. A particularity of the GOST block cipher is that its s-boxes can be secret and they can be used to compose a secondary key, further extending keysize to a total of 610 bits.

The GOST block cipher has been analyzed in [1]–[9] using different techniques. The related-key boomerang attack was proposed in [6]. Nevertheless attack is incorrect; it contains basic ideas to build attacks on the GOST block cipher. So, the attack recovering a 256-bit secret key with 18 related keys was independently developed by V. Rudskoy [7] and the authors.

In this paper related key attacks using 2 and 4 related keys are described. For some classes s-boxes the attacks are practical. We describe two attacks with 4 related keys and one attack using 2 related keys. Our attacks consist of four main steps. In the first step we obtain a set $K^{[1]}$

of 32-th round key candidates. In the second step we get a set $K^{[2]}$ consisting of round key candidates for 27-31 rounds. For finding the set $K^{[2]}$ we apply a related-key truncated-differential attack and the set $K^{[1]}$. In the third step we obtain a set $K^{[3]}$ of 26-th round key candidates using a related-key differential attack and the set $K^{[3]}$. In the fourth step we get a set $K^{[4]}$of 25-th round key candidates using a related-key boomerang attack or the exhaustive method. The essential differences of our attacks consist in step 1; the other steps are the same for all attacks. Depending on the properties of the s-boxes the first step requires 2 or 4 related keys.

We will use the following notations: $x \in_U X$ -x is randomly chosen from the set X; V_n is the n-dimensional vector space over $GF(2)$; $\varepsilon_i = (0, ..., 0, 1, \underbrace{0, ...0}_{i})$, $i = 0, 1, ...$; $s = (s_8, ..., s_1)$ is a nonlinear (s- boxes) layer in the round function f; k_i is a round key of i-th round, $i = 1, ..., 32$; $\alpha^{(j)} = \left(\alpha^{(j,1)}, \alpha^{(j,2)}\right)$ is the ciphertext after j rounds.

2 Ideas of the attacks

To attack with 2 related keys basic principles of the related-key truncated differential attack from [KoLLK04] are used. We consider the pair of related keys k, k' and pairs of plaintexts $\alpha^{(0)}, \alpha'^{(0)}$ such that

$$k \oplus k' = (\varepsilon_{31}, \mathbf{0}, \varepsilon_{31}, \mathbf{0}, \varepsilon_{31}, \mathbf{0}, \varepsilon_{31}, \mathbf{0}, \varepsilon_{31}, \mathbf{0}), \alpha^{(0)} \oplus \alpha'^{(0)} = (\mathbf{0}, \varepsilon_{31}).$$

We use the special technique to classify all bijective 4 × 4-bit s-boxes with regard to the truncated-differential probabilities. This classification allows us to describe some classes of "strong" and "weak"s-boxes. So, if we use strong s-boxes than we cannot mount our attack on the GOST block cipher but we can attack it with weak s-boxes. For weak s-boxes the time complexity T_1 and the number n_1 of texts satisfy the following inequalities $T_1 \leq 2^{49}$ encryptions, $n_1 \leq 2^{32}$ and the success probability is 0.99.

Step 2 is also based on the related-key truncated differential attack. We use a truncated differential characteristic to find round keys $k_{31}, ..., k_{27}$. They can only be recovered for weak s-boxes. Now we discuss two attacks with 4 related keys. Their step 4, which recovers the round key $k^{(32)}$, is based on a related-key boomerang attack and uses related keys

$$k \oplus k' = k'' \oplus k''' = (\varepsilon_{31}, \mathbf{0}, \varepsilon_{31}, \mathbf{0}, \varepsilon_{31}, \mathbf{0}, \boldsymbol{\varepsilon}_{31}, \mathbf{0}, \boldsymbol{\varepsilon}_{31}, \mathbf{0}),$$
$$k \oplus k'' = k' \oplus k''' = (\varepsilon_0, \mathbf{0}, \mathbf{0}, \mathbf{0}, \mathbf{0}, \mathbf{0}, \mathbf{0}, \mathbf{0}).$$

In step 1 of the first attack the related-key boomerang distinguisher from [7] is applied to recover the round key k_{32}. But it can easily be proved that such the distinguisher does not recover k_{32} if the s-box s_8 has the linear translator ε_3, i.e.

$$s_8(\alpha \oplus \varepsilon_3) \oplus s_8(\alpha) = \delta$$

for some $\delta \in V_4$ and all $\alpha \in V_4$. In step 1 of the second attack with 4 related keys we use another related-key boomerang distinguisher to recover k_{32}. This distinguisher can be applied for any set of s-boxes and does not depend on their properties. But there exists a set of "strong" keys for which it is not applicable. The distinguisher is based on a related-key reverse boomerang technique in the chosen ciphertext model as in [7]. We also use the same differential characteristics as in [7]. As input we use pairs of ciphertexts $\left(\alpha^{(32,1)}, \alpha^{(32,2)}\right), \left(\alpha'^{(32,1)}, \alpha'^{(32,2)}\right)$ such that

$$\alpha^{(32,2)} = \alpha'^{(32,2)} = \sum_{i=0}^{j} \varepsilon_i, \alpha'^{(32,1)} = \alpha^{(32,1)} \oplus \lambda_j,$$

where $\alpha^{(32,1)} \in_R V_{32}$, $j \in \{0, ..., 31\}$ and $\lambda_j \in V_{32}$. We start with $j = 0$ and try to find the corresponding $\lambda_j \in V_{32}$, so that we could get a correct quartet for $\left(\alpha^{(32,1)}, \alpha^{(32,2)}\right)$. If we try all $\lambda_j \in V_{32}$, we will obtain the correct quartet with probability 1. Analyzing active s-boxes we could recover 27 bits of round key k_{32}, except the first bit and the last 4 bits. The time complexity in the worst case for keys

$$k = \sum_{i=1}^{27} \varepsilon_i \oplus \sum_{j \in \{1,28,29,30,31\}} \kappa_j \varepsilon_j, \kappa_j \in_R \{0, 1\}$$

can be evaluated as $T_3^{(32)} = 4 \cdot 2^{32}$ GOST encryptions.

The third step uses the same technique as in the first step to recover the round key $k^{(26)}$. We use the key pair k, k' such that

$$k \oplus k' = (\varepsilon_{31}, \mathbf{0}, \varepsilon_{31}, \mathbf{0}, \varepsilon_{31}, \mathbf{0}, \varepsilon_{31}, \mathbf{0}, \varepsilon_{31}, \mathbf{0})$$

to obtain a specific difference in the pair $\left(\alpha^{(26,1)}, \alpha^{(26,2)}\right), \left(\alpha'^{(26,1)}, \alpha'^{(26,2)}\right)$.

The last step uses a related-key boomerang attack with 2 related keys based on the first step. Note that when we apply the second attack with we have a class of strong keys for which the used distinguishers are not applicable.

The time complexity T_2 (the number of encryptions) and the number n_2 of texts of the first attack with 4 related keys can be evaluated as

$2^{30,63} \leq T_2 \leq 2^{60,92}$, $2^{14} \leq n_2 \leq 2^{44}$. The time complexity T_3 (the number of encryptions) and the number n_3 of texts of the second attack with 4 related keys can be evaluated as $2^{35,63} \leq T_3 \leq 2^{65,92}$, $2^{14} \leq n_3 \leq 2^{44}$. For the s-boxes from [10] we have $T_2 = 2^{44.79}$ encryptions, $n_2 = 2^{26.18}$ plaintexts, the success probability is 0.99 but the attack with 2 related keys is not applicable.

References

1. Kelsey J., Schneier B., Wagner D., Key-schedule cryptanalysis of IDEA, G-DES, GOST, SAFER, and triple-DES, In Crypto'96, volume 1109 of Lecture Notes in Computer Science, Springer, (1996)
2. Seki H., Kaneko T., Differential cryptanalysis of reduced rounds of gost. Selected Areas in Cryptography, volume 2012 of Lecture Notes in Computer Science, pages 315-323. Springer, (2000)
3. Biham E., Dunkelman O., Keller N. Improved slide attacks. FSE, volume 4593 of Lecture Notes in Computer Science, pages 153-166. Springer, (2007)
4. Kara O., Reflection Cryptanalysis of Some Ciphers. INDOCRYPT, volume 5365 of Lecture Notes in Computer Science, pages 294-307. Springer, (2008)
5. Ko Y., Hong S., Lee W., Lee S., Kang J.-S., Related key differential attacks on 27 rounds of xtea and full-round gost. FSE, volume 3017 of Lecture Notes in Computer Science, pages 299-316. Springer, (2004)
6. Fleischmann E., Gorski M., Huhne J.-H., Lucks S., Key Recovery Attack on full GOST Block Cipher with Zero Time and Memory, WEWoRC, (2009)
7. Rudskoy V., On zero practical significance of "Key recovery attack on full GOST block cipher with zero time and memory", http://eprint.iacr.org/2010/,)2010)
8. Isobe T., A Single-Key Attack on the Full GOST Block Cipher, In FSE 2011, Fast Software Ecnryption, Springer LNCS, (2011)
9. Courtois N., Security Evaluation of GOST 28147-89 in view of international standardisation, http://eprint.iacr.org/2011/211, (2011)
10. Schneier B., Applied Cryptography, Second Edition, John Wiley Sons, (1996)

On the Security of Hummingbird-2 against Side Channel Cube Attacks

Xinxin Fan and Guang Gong

Department of Electrical and Computer Engineering
University of Waterloo
Waterloo, Ontario, N2L 3G1, CANADA
{x5fan, ggong}@uwaterloo.ca

1 Introduction

The cube attack, proposed by Dinur and Shamir at EUROCRYPT 2009 [1], is a generic key-recovery attack that may be applied to any cryptosystem, provided that the adversary can obtain a bit of information that can be represented by a low-degree decomposition multivariate polynomial in Algebraic Normal Form (ANF) of the secret and public variables of the target cryptosystem. An interesting feature of the cube attack is that it only requires a black-box access to a cryptosystem, that is, the internal structure of the target cryptographic primitive is unknown. Considering the practical implementations of cryptosystems, especially on various embedded devices, Dinur and Shamir [2] also proposed a side channel attack model in which the attacker is assumed to have access to some *limited* information leaked about the internal state of the cryptographic primitive. In this contribution, we investigate the security of the Hummingbird-2 cipher [4] against the cube attack under the single-bit-leakage side channel attack model. Our experimental results show that using a single bit of the internal state during the initialization process of the Hummingbird-2 cipher we can recover the first 48 key bits of the Hummingbird-2.

2 A Review on the Cube Attack

In the cube attack, a cryptographic primitive is viewed as a set of multivariate polynomials $p(v_1, \cdots, v_m, k_1, \cdots, k_n)$ over $\mathbb{F}_2$, each of them mapping m public variables v_i (i.e., plaintext bits in block ciphers and keyed hash functions or initial values in stream ciphers) and n secret variables k_i (i.e., key bits) to one of the ciphertext bits. Let $I = \{i_1, i_2, \ldots, i_k\} \subseteq \{1, 2, \ldots, m\}$ be a subset of the public variable indices and $t_I = x_{i_1} x_{i_2} \cdots x_{i_k}$ be a monomial term. Then the polynomial p, which is called a *master* polynomial, is decomposed as follows:

$$p(v_1, \cdots, v_m, k_1, \cdots, k_n) = t_I \cdot p_{S(I)} + q(v_1, \cdots, v_m, k_1, \cdots, k_n),$$

where t_I is called a *cube* that contains only a subset of public variables and $p_{S(I)}$ is called the *superpoly* of t_I in p. Note that the superpoly of I in p does not contain any common variables with t_I and each monomial in q does not contain at least one variable from I. A term t_I is called a *maxterm* if its superpoly in p is linear polynomial with $\deg(p_{S(I)}) = 1$.

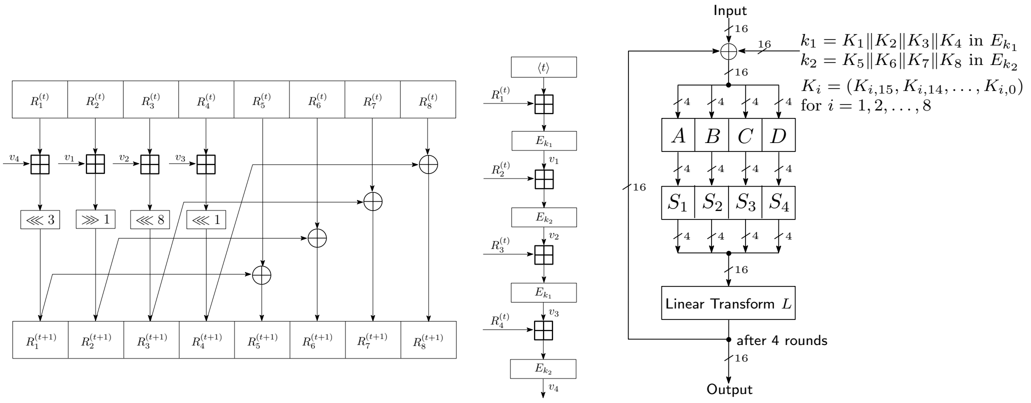
The main observation of the cube attack is that the symbolic sum over $\mathbb{F}_2$ of all evaluations of p by assigning all the possible combinations of 0/1 values to the public variables v_i's with $i \in I$ and fixing the value of all the remaining v_i's with $i \notin I$ is exactly $p_{S(I)}$, the superpoly of t_I in p. The cube attack consists of a pre-processing (offline) and an online phase. While the pre-processing phase aims to find monomials t_I's that lead to linear superpolys, the online phase solves the linear equations obtained from the pre-processing phase to recover the secret key.

3 The Initialization of Hummingbird-2

Hummingbird-2 is a security enhanced version of its predecessor Hummingbird-1 [3], in response to the cryptanalysis work in [5]. The design of Hummingbird-2 adopts the same *hybrid* structure of block cipher and stream cipher as the Hummingbird-1 with 16-bit block size, 128-bit key size, and 128-bit internal state. To launch the cube attack, we solely focus on the initialization process of the Hummingbird-2 as shown in Fig. 1(a), where $\boxplus$ denotes an addition modulo 2^{16}, $\oplus$ an exclusive-OR operation, and $\lll$ (or $\ggg$) a left (or right) circular shift operation, respectively. The initialization process consists of four 16-bit block ciphers E_{k_i} $(i = 1, 2)$ and eight 16-bit internal state registers $R_i^{(t)}$ $(i = 1, \ldots, 8$ and $t = 0, 1, \ldots)$. Initially, the register $R_i^{(0)}$ is set as follows:

$$R_i^{(0)} = \begin{cases} \mathsf{NONCE}_i & \text{for } i = 1, 2, 3, 4 \\ \mathsf{NONCE}_{i \bmod 4} & \text{for } i = 5, 6, 7, 8 \end{cases},$$

where NONCE_i $(i = 1, \ldots, 4)$ is the i-th 16-bit nonce. The 128-bit secret key K is divided into two 64-bit subkeys k_1 and k_2 which are used in the four block ciphers E_{k_i} $(i = 1, 2)$, respectively.


$R_1^{(t)}$ $R_2^{(t)}$ $R_3^{(t)}$ $R_4^{(t)}$ $R_5^{(t)}$ $R_6^{(t)}$ $R_7^{(t)}$ $R_8^{(t)}$
v_4 v_1 v_2 v_3
$\lll 3$ $\ggg 1$ $\lll 8$ $\lll 1$
$R_1^{(t+1)}$ $R_2^{(t+1)}$ $R_3^{(t+1)}$ $R_4^{(t+1)}$ $R_5^{(t+1)}$ $R_6^{(t+1)}$ $R_7^{(t+1)}$ $R_8^{(t+1)}$
$\langle t \rangle$
E_{k_1} E_{k_2} E_{k_1} E_{k_2}
Input
16
$k_1 = K_1\|K_2\|K_3\|K_4$ in E_{k_1}
$k_2 = K_5\|K_6\|K_7\|K_8$ in E_{k_2}
$K_i = (K_{i,15}, K_{i,14}, \ldots, K_{i,0})$
for $i = 1, 2, \ldots, 8$
4
A B C D
S_1 S_2 S_3 S_4
Linear Transform L
after 4 rounds
Output

(a) Initialization Process

(b) 16-bit Block Cipher $E_{k_i}, i = 1, 2$

Fig. 1. Hummingbird-2 Initialization and Building Blocks

The Hummingbird-2 initialization employs four identical block ciphers $E_{k_i}(\cdot)$ $(i = 1, 2)$ in a consecutive manner, each of which is a typical substitution-permutation (SP) network with 16-bit block size and 64-bit key as shown in Fig. 1(b). The block cipher consists of four rounds, each of which is comprised of a key mixing step, a substitution layer, and a permutation layer. The key mixing step is implemented using a simple exclusive-OR operation, whereas the substitution layer is composed of four S-boxes $4-8$ of the Serpent block cipher with 4-bit inputs and 4-bit outputs. The permutation layer in the 16-bit block cipher is given by the linear transform $L : \{0, 1\}^{16} \to \{0, 1\}^{16}$ defined as follows:

$$L(x) = x \oplus (x \lll 6) \oplus (x \lll 10),$$

where $x = (x_{15}, x_{14}, \ldots, x_0)$ is a 16-bit data block. The 64-bit subkeys k_i $(i = 1, 2)$ are split into four 16-bit round keys (see Figure 1(b)) that are used in the four rounds, respectively. The entire initialization process consists of four rounds and after each round the eight internal state registers are updated from $R_i^{(t)}$ to $R_i^{(t+1)}$ $(i = 1, \ldots, 8)$ in an unpredictable way based on their current states as well as the outputs of the 16-bit block ciphers. For more details about the Hummingbird-2 cipher, the interested reader is referred to [4].

For applying the single-bit-leakage side channel cube attack to the Hummingbird-2 initialization process, we assume that there is a single bit leakage after the third round of the first 16-bit block cipher E_{k_1}. This enables us to recover the first 48 bits of the secret key K in the Hummingbird-2. As an example of the attack, we provide the analysis results for the least significant bit of the internal state after the third round of E_{k_1} in this contribution.

4 Linearity and Quadraticity Tests

Let $\mathbb{F}_2^n$ be an n-dimensional vector space over the finite field $\mathbb{F}_2$ and $f(x_1, \ldots, x_n) : \mathbb{F}_2^n \to \mathbb{F}_2$ be a Boolean function of n variables. One of the crucial steps in the cube attack is to test whether the

Boolean function f is linear or quadratic, and if so, we need to find out the expression of f. More specifically, we need to consider the following two cases:

- A Boolean function f is linear in its inputs if it satisfies $f(x \oplus y) = f(x) \oplus f(y) \oplus f(0)$ for all $x, y \in \mathbb{F}_2^n$. Such a linear function has a form of $f(x_1, \ldots, x_n) = \bigoplus_{1 \le i \le n} a_i x_i \oplus a_0$, where $a_i \in \mathbb{F}_2$ and $a_0 = f(0)$.
- A Boolean function f is quadratic in its inputs if it satisfies $f(x \oplus y \oplus z) = f(x \oplus y) \oplus f(x \oplus z) \oplus f(y \oplus z) \oplus f(x) \oplus f(y) \oplus f(z) \oplus f(0)$ for all $x, y, z \in \mathbb{F}_2^n$. Such a quadratic function has a form of $f(x_1, \ldots, x_n) = \bigoplus_{1 \le i < j \le n} a_{ij} x_i x_j \oplus \bigoplus_{1 \le i \le n} a_i x_i + a_0$, where $a_{ij}, a_i \in \mathbb{F}_2$ and $a_0 = f(0)$.

In [6], Zhu *et al.* proposed an efficient *term-by-term* linearity test (see Algorithm 1 in Fig. 2) in which the Boolean function f needs to be evaluated $n + 1 + 2 \cdot d_1 \cdot C_1$ times in order to discover the linear secret variables within a superpoly equation and test their linearity, where n is the number of secret variables, d_1 is the number of linear terms, and C_1 is the total number of linearity tests. We generalize this approach to the quadratic case and obtain a faster term-by-term quadraticity test (see Algorithm 2 in Fig. 2). The basic idea is to first find all linear and quadratic terms in the superpoly equation, followed by a probabilistic linearity (and/or quadraticity) test for each individual linear (and/or quadratic) term.

```
Algorithm 1 Term-by-Term Linearity Test [7]
$S_l \leftarrow$ an empty set
for $i \leftarrow 1$ to $n$ do
    $x \leftarrow (0, 0, \ldots, 1, \ldots, 0)$ where only the $i$-th bit is 1
    if $f(x) \oplus f(0) = 1$ then
        Add $i$ into the set $S_l$
    end if
end for
for each $j \in S_l$ do
    for $c \leftarrow 1$ to $C_1$ do
        Randomly choose an input value $y \in \mathbb{F}_2^n$
        $y' \leftarrow y$
        $y_j \leftarrow 0$ and $y'_j \leftarrow 1$
        if $f(y) = f(y')$ then
            Reject and halt
        end if
    end for
end for
```

```
Algorithm 2 Term-by-Term Quadraticity Test
$S_l \leftarrow$ an empty set
$S_q \leftarrow$ an empty set
Execute the term-by-term linearity test (see the Algorithm 1)
for $i \leftarrow 1$ to $n - 1$ do
    for $j \leftarrow i + 1$ to $n$ do
        $x \leftarrow (0, \ldots, 1, \ldots, 1, \ldots, 0)$ where only the $i$-th and $j$-th bits are 1
        if $(f(x) \oplus f(0) = 1$ and $(i, j \in S_l$ or $i, j \notin S_l))$ or
            $(f(x) \oplus f(0) = 0$ and $(i \in S_l, j \notin S_l$ or $i \notin S_l, j \in S_l)))$ then
                Add $(i, j)$ into the set $S_q$
        end if
    end for
end for
for each $(i, j) \in S_q$ do
    for $c \leftarrow 1$ to $C_2$ do
        Randomly choose an input value $y^{(1)} \in \mathbb{F}_2^n$
        $y^{(2)} \leftarrow y^{(1)}, y^{(3)} \leftarrow y^{(1)}$, and $y^{(4)} \leftarrow y^{(1)}$
        $(y_i^{(1)}, y_j^{(1)}) \leftarrow (0, 0)$ and $(y_i^{(2)}, y_j^{(2)}) \leftarrow (0, 1)$
        $(y_i^{(3)}, y_j^{(3)}) \leftarrow (1, 0)$ and $(y_i^{(4)}, y_j^{(4)}) \leftarrow (1, 1)$
        if $f(y^{(1)}) \oplus f(y^{(2)}) = f(y^{(3)}) \oplus f(y^{(4)})$ then
            Reject and halt
        end if
    end for
end for
```

Fig. 2. The Term-by-Term Linearity and Quadraticity Tests

Using the above term-by-term quadratic test, the Boolean function f needs to be totally evaluated $(n + 1 + 2 \cdot d_1 \cdot C_1) + \left(\frac{n(n-1)}{2} + 4 \cdot d_2 \cdot C_2 \right)$ times for a superpoly with d_1 linear terms and d_2 quadratic terms, where C_1 and C_2 are the number of linearity and quadraticity tests, respectively.

5 Side Channel Cube Attack on Hummingbird-2

As shown in Fig. 1(b) the 64-bit subkey k_1 in the block cipher E_{k_1} is divided into four 16-bit round keys $K_i = (K_{i,15}, \ldots, K_{i,0}), i = 1, 2, 3, 4$. At the i-th round the round key K_i is exclusive-ORed with the internal state of the block cipher E_{k_1}. Hence, after the third round, the internal state of the block cipher E_{k_1} contains the information about the round keys K_1, K_2 and K_3. In order to find the maxterms from a master polynomial associated with the least significant bit (LSB)[1] of

[1] The cube attack can also be applied to any other single bit of the internal state after the third round of the block cipher E_{k_1} in a straightforward way.

the internal state after the third round, we exhaustively test all possible cube sizes ranging from 1 to 16 (Recall that in the first round of the Hummingbird-2 initialization the input to the first block cipher E_{k_1} is the 16-bit nonce NONCE_1). Moreover, we fully exploit the power of GPU (i.e., a GeForce GTX 285 graphics card from NVIDIA) to significantly accelerate the evaluation of the master polynomial by launching 2^κ (κ is the size of a cube) threads simultaneously, each of which calculates the value of the master polynomial for one of 2^κ different 0/1 combinations of a subset of public variables $(v_{i_1}, v_{i_2}, \ldots, v_{i_\kappa})$.

Table 1. The Cube Indices and the Superpoly Equations for the Hummingbird-2 Initialization from the Least Significant Bit Leakage after the Third Round of the First E_{k_1}

Cube Indices	Cube Size	Linear Superpoly Equation
$\{11, 10, 9, 8, 7, 6, 5, 4, 3, 2\}$	10	$K_{1,0} + K_{1,1} + 1$
$\{15, 14, 13, 12, 7, 6, 5, 4, 3, 0\}$	10	$K_{1,1}$
$\{11, 10, 9, 8, 7, 6, 5, 4, 3, 0\}$	10	$K_{1,2} + 1$
$\{11, 10, 9, 8, 7, 6, 5, 4, 2, 0\}$	10	$K_{1,3}$
$\{11, 10, 9, 8, 7, 3, 2, 1, 0\}$	9	$K_{1,4}$
$\{15, 14, 13, 12, 11, 10, 9, 8, 6, 4\}$	10	$K_{1,5}$
$\{15, 14, 13, 12, 4, 3, 2, 1, 0\}$	9	$K_{1,5} + K_{1,6}$
$\{11, 10, 9, 8, 4, 3, 2, 1, 0\}$	9	$K_{1,7}$
$\{15, 14, 13, 12, 10, 9, 3, 2, 1, 0\}$	10	$K_{1,8}$
$\{11, 8, 7, 6, 5, 4, 3, 2, 1, 0\}$	10	$K_{1,9}$
$\{15, 14, 13, 12, 9, 8, 3, 2, 1, 0\}$	10	$K_{1,10}$
$\{15, 14, 13, 12, 9, 8, 7, 6, 5, 4\}$	10	$K_{1,11}$
$\{14, 13, 7, 6, 5, 4, 3, 2, 1, 0\}$	10	$K_{1,12}$
$\{15, 12, 7, 6, 5, 4, 3, 2, 1, 0\}$	10	$K_{1,13} + K_{1,14}$
$\{15, 12, 11, 10, 9, 8, 7, 6, 5, 4\}$	10	$K_{1,14}$
$\{15, 12, 11, 10, 9, 8, 7, 6, 5, 4\}$	10	$K_{1,15} + 1$
$\{14, 13, 12, 10, 9, 8, 7, 6, 4, 3, 2, 1, 0\}$	13	$K_{2,0} + K_{2,5} + K_{2,6} + K_{2,7} + K_{2,9} + K_{2,11} + K_{2,13}$
$\{14, 13, 12, 11, 9, 8, 7, 6, 4, 3, 2, 1, 0\}$	13	$K_{2,3} + K_{2,5} + K_{2,6} + K_{2,7} + K_{2,10} + K_{2,11} + K_{2,13} + 1$
$\{14, 13, 12, 11, 10, 8, 7, 6, 4, 3, 2, 1, 0\}$	13	$K_{2,2} + K_{2,3} + K_{2,6} + K_{2,7} + K_{2,8} + K_{2,10} + K_{2,11} + K_{2,13} + K_{2,14} + K_{2,15} + 1$
$\{14, 13, 12, 11, 10, 9, 7, 6, 4, 3, 2, 1, 0\}$	13	$K_{2,0} + K_{2,2} + K_{2,3} + K_{2,5} + K_{2,8} + K_{2,9} + K_{2,10} + K_{2,14} + K_{2,15}$
$\{14, 13, 12, 11, 10, 9, 8, 7, 6, 4, 2, 1, 0\}$	13	$K_{2,2} + K_{2,3} + K_{2,4} + K_{2,5} + K_{2,7} + K_{2,9} + K_{2,10} + K_{2,11} + K_{2,13} + K_{2,14}$
$\{14, 13, 12, 11, 10, 9, 8, 7, 6, 4, 3, 1, 0\}$	13	$K_{2,5} + K_{2,8} + K_{2,11} + K_{2,14}$
$\{14, 13, 12, 11, 10, 9, 8, 7, 6, 4, 3, 2, 0\}$	13	$K_{2,0} + K_{2,10} + K_{2,12}$
$\{15, 13, 12, 10, 9, 8, 7, 6, 4, 3, 2, 1, 0\}$	13	$K_{2,5} + K_{2,7} + K_{2,9} + K_{2,11} + K_{2,13} + 1$
$\{15, 13, 12, 11, 9, 8, 7, 6, 4, 3, 2, 1, 0\}$	13	$K_{2,0} + K_{2,1} + K_{2,3} + K_{2,4} + K_{2,5} + K_{2,10} + K_{2,11} + K_{2,13}$
$\{15, 13, 12, 11, 10, 8, 7, 6, 4, 3, 2, 1, 0\}$	13	$K_{2,1} + K_{2,2} + K_{2,3} + K_{2,4} + K_{2,6} + K_{2,8} + K_{2,10} + K_{2,11} + K_{2,13} + K_{2,14} + K_{2,15}$
$\{15, 13, 12, 11, 10, 9, 7, 6, 4, 3, 2, 1, 0\}$	13	$K_{2,1} + K_{2,2} + K_{2,3} + K_{2,4} + K_{2,5} + K_{2,6} + K_{2,8} + K_{2,9} + K_{2,10} + K_{2,14} + K_{2,15}$
$\{15, 13, 12, 11, 10, 9, 8, 7, 6, 4, 2, 1, 0\}$	13	$K_{2,1} + K_{2,2} + K_{2,3} + K_{2,4} + K_{2,5} + K_{2,10} + K_{2,12} + K_{2,13} + K_{2,14} + K_{2,15} + 1$
$\{15, 13, 12, 11, 10, 9, 8, 7, 6, 4, 3, 1, 0\}$	13	$K_{2,1} + K_{2,2} + K_{2,5} + K_{2,6} + K_{2,7} + K_{2,9} + K_{2,12} + K_{2,14} + K_{2,15} + 1$
$\{15, 13, 12, 11, 10, 9, 8, 7, 6, 4, 3, 2, 0\}$	13	$K_{2,0} + K_{2,3} + K_{2,5} + K_{2,6} + K_{2,9} + K_{2,10} + K_{2,11} + 1$
$\{15, 14, 12, 11, 9, 8, 7, 6, 4, 3, 2, 1, 0\}$	13	$K_{2,0} + K_{2,1} + K_{2,3} + K_{2,4} + K_{2,6} + K_{2,7} + K_{2,9}$
$\{15, 14, 12, 11, 10, 9, 8, 7, 6, 4, 2, 1, 0\}$	13	$K_{2,1} + K_{2,2} + K_{2,3} + K_{2,8} + K_{2,11} + K_{2,13} + K_{2,15}$

After running the faster term-by-term quadratic test (see the Algorithm 2 in Section 4) on a single PC (with a GPU) for a few days, we have been able to find tens of linear and quadratic superpoly equations using different cube sizes. For the linear superpolys, we use the Gaussian elimination to remove the linear dependent equations and obtain 32 linear independent equations with 32 key variables as a result. Table 1 lists the indices of the public variables in the maxterms and the corresponding linear superpoly equations. For the quadratic superpolys, we find that they are all redundant and cannot provide more information about the secret key bits. As shown in Table 1, we have 3 maxterms of size 9, 13 maxterms of size 10, and 16 maxterms of size 13. Therefore, in order to recover the first 32 key bits of the secret key, the total number of chosen plaintexts (i.e., the nonce NONCE_1) at the online phase of the cube attack is $3 \times 2^9 + 13 \times 2^{10} + 16 \times 2^{13} \approx 2^{17.155}$. After obtaining the first 32 key bits K_1 and K_2, we can use those key bits to significantly simplify the master polynomial associated with the LSB of the internal state of the block cipher E_{k_1} after the third round. As an example, we assume that $K_1 = \mathsf{0x35df}$ and $K_2 = \mathsf{0xac2b}$. Using the Algorithm 2, we find 12 linear independent equations (see Table 2) with 12 secret key variables $K_{3,4}, K_{3,5}, \ldots, K_{3,15}$, which enable us to solve those key variables at the online phase with $2^4 + 2 \times 2^5 + 6 \times 2^6 + 3 \times 2^7 \approx 2^{9.755}$ chosen plaintexts. The remaining four secret key bits $K_{3,0}, K_{3,1}, K_{3,2}$ and $K_{3,3}$ can be obtained by conducting an exhaustive search. Moreover, one can also obtain the relations among those four key bits by solving the quadratic equations in Table 2 with another $2 \times 2^5 = 64$ chosen plaintexts. Consequently, the total time complexity to find the correct 128-bit secret key of the Hummingbird-2 has been reduced to $O(2^{80})$.

Table 2. The Cube Indices and the Superpoly Equations for the Hummingbird-2 Initialization from the Least Significant Bit Leakage after the Third Round of the First E_{k_1} (K_1 and K_2 are known)

Cube Indices	Cube Size	Superpoly Equation
$\{7,6,5,4\}$	4	$K_{3,4} + K_{3,6} + K_{3,7} + K_{3,8} + K_{3,10} + K_{3,11} + K_{3,12} + K_{3,14} + K_{3,15} + 1$
$\{7,6,5,4,0\}$	5	$K_{3,4} + 1$
$\{12,10,7,4,0\}$	5	$K_{3,12} + 1$
$\{11,9,5,4,3,0\}$	6	$K_{3,10}$
$\{11,10,9,8,6,4\}$	6	$K_{3,13}$
$\{7,6,5,4,3,0\}$	6	$K_{3,6} + K_{3,7} + K_{3,12}$
$\{6,4,3,2,1,0\}$	6	$K_{3,4} + K_{3,9} + K_{3,10} + K_{3,11} + 1$
$\{11,10,9,6,4,0\}$	6	$K_{3,4} + K_{3,5} + K_{3,8} + K_{3,10} + K_{3,15} + 1$
$\{9,8,3,2,1,0\}$	6	$K_{3,5} + K_{3,7} + K_{3,8} + K_{3,11} + K_{3,14} + K_{3,15} + 1$
$\{8,6,5,3,2,1,0\}$	7	$K_{3,6} + K_{3,7} + K_{3,8} + K_{3,9}$
$\{8,7,6,3,2,1,0\}$	7	$K_{3,6} + K_{3,7} + K_{3,8} + K_{3,10} + K_{3,11} + 1$
$\{8,7,6,5,4,2,1\}$	7	$K_{3,5} + K_{3,6} + K_{3,8} + K_{3,9} + K_{3,10} + K_{3,13} + K_{3,14} + K_{3,15}$
$\{10,8,7,6,4\}$	5	$K_{3,7} + K_{3,8} + K_{3,9} + K_{3,11} + K_{3,12} + K_{3,13} + K_{3,14} + K_{3,8}K_{3,11} + K_{3,9}K_{3,11} + K_{3,4}K_{3,5} + K_{3,5}K_{3,6} + K_{3,2}K_{3,3}$
$\{9,8,6,5,2\}$	5	$K_{3,4} + K_{3,7} + K_{3,9} + K_{3,10} + K_{3,12} + K_{3,9}K_{3,11} + K_{3,10}K_{3,11} + K_{3,5}K_{3,7} + K_{3,4}K_{3,5} + K_{3,1}K_{3,2} + K_{3,0}K_{3,2}$

6 Conclusions

In this contribution we investigate the security of the Hummingbird-2 against the side channel cube attacks under the single-bit-leakage model. Our experimental results show that one can recover the first 48 bits of the secret key in the Hummingbird-2, by taking advantage of a single bit information leakage from the internal state after the third round of the first block cipher E_{k_1}. The data complexity of the proposed attack is around 2^{18}. Moreover, an efficient term-by-term quadratic test is also proposed. Finally, we would like to point out that in order to launch the side channel cube attack against the Hummingbird-2 an attacker needs to acquire the exact value of the least significant bit after the third round of the block cipher E_{k_1}, which is, if not impossible, quite difficult and expensive in practice according to the current manufacturing technology of embedded systems. Therefore, the proposed attack is only of a theoretical interest at the moment and does not directly jeopardize the security of the Hummingbird-2 implementations in practice.

References

1. I. Dinur and A. Shamir, "Cube Attacks on Tweakable Black Box Polynomials", *Advances in Cryptology - EUROCRYPT 2009*, LNCS 5479, A. Joux (ed.), Berlin, Germany: Springer-Verlag, pp. 278-299, 2009.
2. I. Dinur and A. Shamir, "Side Channel Cube Attacks on Block Ciphers", Cryptology ePrint Archive, Report 2009/127 (2009), `http://eprint.iacr.org/2009/127`.
3. D. Engels, X. Fan, G. Gong, H. Hu, and E. M. Smith, "Hummingbird: Ultra-Lightweight Cryptography for Resource-Constrained Devices", *FC 2010 Workshops, RLCPS, WECSR, and WLC 2010*, ser. LNCS 6054, R. Curtmola et al. (eds.), Berlin, Germany: Springer-Verlag, pp. 3-18, 2010.
4. D. Engels, M.-J. O. Saarinen, and E. M. Smith, "The Hummingbird-2 Lightweight Authenticated Encryption Algorithm", to appear in the proceedings of *The 7th Workshop on RFID Security and Privacy - RFIDSec 2011*, Berlin, Germany: Springer-Verlag, 2011.
5. M.-J. O. Saarinen, "Cryptanalysis of Hummingbird-1", *The 18th International Workshop on Fast Software Encryption - FSE 2011*, ser. LNCS 6733, A. Joux (ed.), Berlin, Germany: Springer-Verlag, pp. 328-341, 2011.
6. B. Zhu, W. Yu, and T. Wang, "A Practical Platform for Cube-Attack-Like Cryptanalyses", Cryptology ePrint Archive, Report 2010/644 (2010), `http://eprint.iacr.org/2010/644`.