Tatra Mt. Math. Publ. **33** (2006), 159–168

TATRA MOUNTAINS
Mathematical Publications

A NEW MODE OF USING ALL-OR-NOTHING TRANSFORMS

Valér Čanda — Tran van Trung

ABSTRACT. We suggest a new mode of using all-or-nothing transforms in conjunction with a symmetric encryption scheme. Our goal is to have a secure encryption mode with the property that brute force attacks on the scheme are slowed down by a predefined factor. Unlike [R. L. Rivest: *All-or-nothing encryption and the package transform*, Fast Software Encryption '97, Lecture Notes in Comput. Sci, Vol. 1267, Springer-Verlag, Berlin, 1997, pp. 210–218], we use a different method which is more efficient and flexible, and ensures that an adversary does not even know the exact ciphertext that has to be attacked.

1. Introduction

The concept of an *All-or-Nothing Transform* (AONT) was introduced by Rivest [1]. An AONT is an unkeyed transformation ϕ mapping a sequence of input blocks $(x_1, x_2, \dots, x_s)$ to a sequence of output blocks $(y_1, y_2, \dots, y_{s'})$ having the following properties:

- given all $(y_1, y_2, \dots, y_{s'})$ it is easy to compute $(x_1, x_2, \dots, x_s)$,
- if any one of the y_j is missing, then it is computationally infeasible to obtain any information about any x_i.

Possible applications of AONT include improving security and speed of encryption, reduction of communication volume, construction of exposure resilient functions, and usage in the chaffing and winnowing techniques (see, e.g., [3] and [4] for more references). In this paper we will address the usage of AONT for improving security of symmetric encryption based on block ciphers.

Conventional encryption modi for a block cipher, such as ECB, CBC, or OFB, which transform a sequence of plaintext blocks $(x_1, x_2, \dots, x_s)$ into a sequence of ciphertext blocks $(z_1, z_2, \dots, z_{s'})$, make it possible to obtain a specific plaintext block by performing a single decryption operation. Such encryption modi are called *separable*. Rivest [1] proposed an encryption mode where an adversary

2000 Mathematics Subject Classification: 68P25.
Keywords: all-or-nothing transform, encryption mode, block cipher, w-slow one-way function.

must decrypt *all* ciphertext blocks $(z_1, z_2, \dots, z_{s'})$ before he can obtain *any particular* plaintext block x_i. An encryption mode having this property is said to be *non-separable*. A brute force attack on a non-separable encryption mode is s' times slower than an attack on the corresponding separable mode. In other words, by using non-separable modes one can achieve a *higher security level without extending the key length* of the cipher. This is particularly useful, when the key length of the involved block cipher is considered as marginal, and cannot be extended for technical or legal reasons.

Rivest designed a non-separable mode by composing an AONT with an ordinary encryption mode in the way shown in Figure 1.

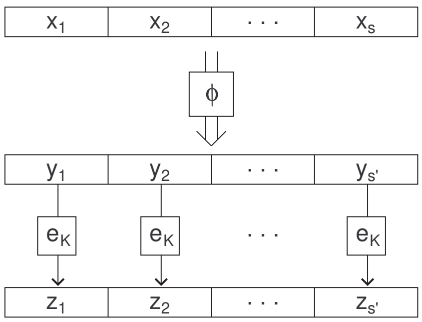

FIGURE 1. AON ECB mode.

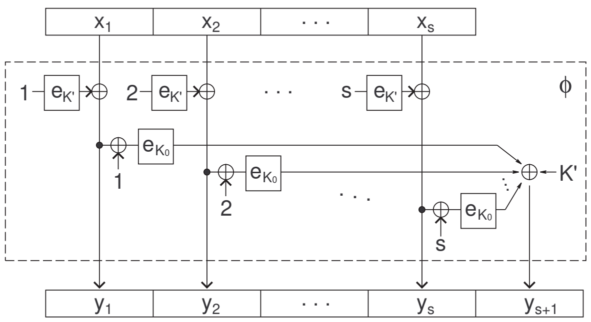

FIGURE 2. AONT construction by Rivest.

In this case the AONT is used as a preprocessing step before the encryption. As the AONT operates directly on the plaintext, it must be a *randomizing* transformation. If it were not, a chosen plaintext attack might yield to a known pseudo-message y_j, which would destroy the non-separability property [1]. Rivest proposes an AONT construction based on a block cipher, as shown in Figure 2. In this construction a randomly chosen key K' is first used for encrypting the message in the CTR mode. Consequently, a randomized hash value of the encrypted message is used for masking K', and the resulting block is appended to the message. The constant K_0, which is used for computing the hash, is publicly-known. Such an AONT is *expanding* (it enlarges the message by one block) and *rather slow* (it needs roughly two encryption operations per block). The complete AON ECB mode (as shown in Figure 1) based on this construction is about three times slower than a simple ECB mode.

Another disadvantage of this approach is the fact that the security gain is given directly by the message length. If we want to ensure a security improvement by factor, say, 2^{10}, we have to pad every (even a very short) message to a total length of 2^{10} blocks. Significant security improvements (e.g., by factor 2^{20} or 2^{30}) are hardly realizable in this way, because the very long resulting messages would slow down any processing or transmission, and would increase the memory requirements.

In what follows we propose a new construction which is noticeably faster than Rivest's design, and provides a custom security gain even for short messages.

2. Our construction

In contrast to Rivest's proposal, we suggest applying an AONT *after* encryption. The advantage of this approach is the fact that the input of ϕ is strongly randomized, and thus ϕ itself does not necessarily need to be a randomizing transform. As a consequence, we can use simpler ϕ constructions which will significantly improve the processing speed.

Our scheme is displayed in Figure 3.

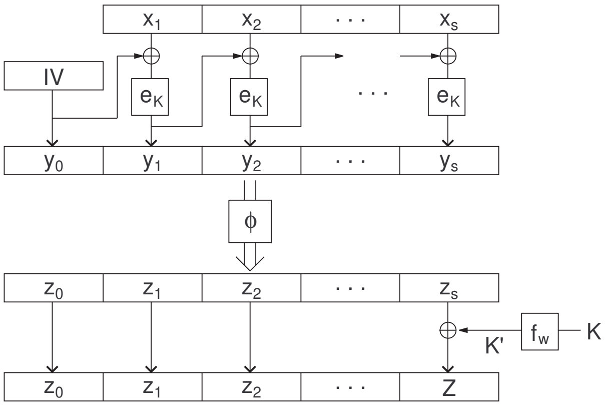

FIGURE 3. OUR AON CBC.

$$M^{-1} = \begin{pmatrix} 1 & 1 & 0 & \cdots & 0 \\ 0 & 1 & 1 & \cdots & 0 \\ 0 & 0 & 1 & \cdots & 0 \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & 0 & \cdots & 1 \\ \lambda & 0 & 0 & \cdots & 1 \end{pmatrix}.$$

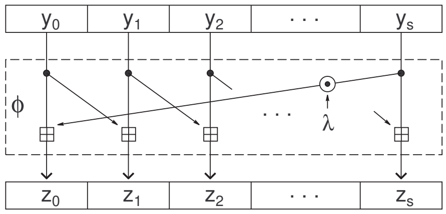

FIGURE 4. OUR AON construction.

In the first step the message $x = (x_1, x_2, \ldots, x_s)$ is encrypted in the regular CBC mode with a randomly chosen initialization vector IV, in the second step the intermediate result $y = (y_0, y_1, \ldots, y_s)$ is transformed by a Stinson-like linear AONT into $z = (z_0, z_1, \ldots, z_s)$, and in the third step the last block z_s is masked by a pseudorandom constant K', which has been derived from the secret key K. Let us now discuss the particular steps in more details.

2.1 Encryption

The purpose of the first step is *encrypting* and *randomizing* the data. Optionally, the CFB mode might be used for that purpose as well. When the underlying cipher is strong we may assume that the intermediate encryption output $y = (y_0, y_1, \ldots, y_s)$ is a cryptographically strong pseudorandom sequence, i.e., it is computationally infeasible to distinguish y from a sequence of uniformly distributed random bits. Because a new random IV is used for every encryption, a repeated processing of the same x will result into different and unpredictable y's.

2.2 AONT

In the second step we use a Stinson-like linear AONT. Stinson [2] proposed the following AONT construction: Let $y = (y_0, y_1, \dots, y_s)$ be a message consisting of blocks $y_i \in \mathbb{F}_q$, where $\mathbb{F}_q$ is a finite field of order $q > 2$. AONT transform $\phi(y)$ is defined as $y \cdot M^{-1}$, where M is an invertible $(s+1) \times (s+1)$ matrix with entries from $\mathbb{F}_q$, such that no entry is equal to zero. Our AONT design together with the appropriate generating matrix M^{-1} is shown in Figure 4. The corresponding matrix M has one of the following two forms, depending on the value of s:

$$M_1 = \frac{1}{\lambda - 1} \cdot \begin{pmatrix} -1 & 1 & -1 & \cdots & 1 & -1 \\ \lambda & -1 & 1 & \cdots & -1 & 1 \\ -\lambda & \lambda & -1 & \cdots & 1 & -1 \\ \vdots & \vdots & \vdots & \ddots & \vdots & \vdots \\ -\lambda & \lambda & -\lambda & \cdots & -1 & 1 \\ \lambda & -\lambda & \lambda & \cdots & \lambda & -1 \end{pmatrix},$$

$$M_2 = \frac{1}{\lambda + 1} \cdot \begin{pmatrix} 1 & -1 & 1 & \cdots & -1 & 1 \\ \lambda & 1 & -1 & \cdots & 1 & -1 \\ -\lambda & \lambda & 1 & \cdots & -1 & 1 \\ \vdots & \vdots & \vdots & \ddots & \vdots & \vdots \\ \lambda & -\lambda & \lambda & \cdots & 1 & -1 \\ -\lambda & \lambda & -\lambda & \cdots & \lambda & 1 \end{pmatrix}.$$

For s odd $M = M_1$ and for s even $M = M_2$. In both cases M contains no zero elements. Note that the value of $\lambda \in \mathbb{F}_q^*$ must be different from $+1$ and -1 in order that the generating matrix is invertible. The operations $+$ and $\cdot$ denote the addition and multiplication in $\mathbb{F}_q$.

Given $y = (y_0, y_1, \dots, y_s)$, one can compute $z = \phi(y)$ as follows:

1. For $i = 1, \dots, s$ compute $z_i = y_i + y_{i-1}$.
2. Compute $z_0 = y_0 + \lambda \cdot y_s$.

The inverse transformation $y = \phi^{-1}(z)$ can be efficiently computed as follows:

1. If s is odd, compute $y_0 = \frac{1}{1-\lambda} \cdot \Big(z_0 + \lambda \cdot \sum\limits_{i=1}^{s} \big((-1)^i \cdot z_i\big)\Big)$,

   otherwise compute $y_0 = \frac{1}{1+\lambda} \cdot \Big(z_0 - \lambda \cdot \sum\limits_{i=1}^{s} \big((-1)^i \cdot z_i\big)\Big)$.

2. For $i = 1, \ldots, s$ compute $y_i = z_i - y_{i-1}$.

Note that if q is a power of two (which is usually the case), there is no distinction between the operations $+$ and $-$, both can be implemented by a binary XOR. Consequently, one does not need to distinguish between even and odd s during the inverse transform, and the factors $(-1)^i$ can be disregarded as well. Hence the first step of the inverse transform will be simplified to $y_0 = \frac{1}{1 \oplus \lambda} \cdot (z_0 \oplus \lambda \cdot \sum_{i=1}^{s} z_i)$. Also the restrictions regarding the value λ will be reduced to $\lambda \neq 1$.

In the AONT construction presented in [2] all the blocks $(y_0, \ldots, y_{s-1})$ are xor-ed with the same value y_s. We use a different setup, as we prefer some more complex dependencies between y and z. In our AONT every single block of y is xor-ed with a different value. For any known continuous subsequence $(z_i, z_{i+1}, \ldots, z_{i+m})$, $m < s$, one cannot obtain the corresponding $(y_i, y_{i+1}, \ldots, y_{i+m})$ because of some missing y_j, $j \notin \{i, \ldots, i+m\}$. The index j can take on any of the values $0, \ldots, s$, depending on the particular combination of i and m. In the AONT construction given by [2] one always misses the same value y_s, independently of i and m. In our opinion, a chain of block dependencies present in our AONT gives a better protection against a partial leakage of y.

2.3 Masking

The function $f_w \colon \{0,1\}^n \longrightarrow \{0,1\}^n$, used for computing K' in the third step of our construction, will be called a *w-slow one-way function*. It must fulfill the following two properties:

1. given any $x \in \{0,1\}^n$, the computation of $y = f_w(x)$ is at least w-times slower than one execution of the encryption function e_K,
2. given any $y \in \{0,1\}^n$, it is computationally infeasible to compute $x = f_w^{-1}(y)$.

We propose f_w as follows: $f_w(K) = K_{w+1}$, where K_i is computed according to the following recurrence: $K_0 = 0$, $K_1 = K$, and $K_i = e_{K_{i-1}}(K_{i-1} \oplus K_{i-2})$ for $i > 1$. Many other designs are certainly possible. One can, for instance, define several f_q's based on principles that are used in one-way hash function design based on block ciphers (see, e.g., [5. Sec. 18.11]).

3. Security

Our mode is not non-separable in the sense defined by Rivest, as one does not necessarily need to decrypt all ciphertext blocks in order to obtain a particular plaintext block. However, our mode is non-separable in the sense that, knowing the proper key K and all ciphertext blocks $(z_0, z_1, \dots, z_s)$, one still has to perform at least w encryption operations to obtain any plaintext block. This means that we ensure the actual goal of a non-separable mode (which is slowing down a brute force attack by a given factor) using a different setup.

In contrast to computational security of Rivest's AONT, Stinson's AONT construction is unconditionally secure. This means one can *prove* that if any n-bit output block z_j is unknown, then any input block y_i can take on every of the 2^n possible values, depending on the value of z_j. On the other hand, because of its linearity and non-randomizing character, Stinson's AONT should not be applied directly on a plaintext (as proposed, e.g., in Figure 1). Even if all particular input blocks are completely protected, an adversary can still obtain information about linear dependencies between several input blocks. This kind of information could be used in combination with a chosen-plaintext attack to defeat the non-separability property. In our construction (Figure 3) we apply the linear AONT on a strongly randomized input $(y_0, y_1, \dots, y_s)$. Because of the random IV used in the CBC mode the blocks $(y_0, y_1, \dots, y_s)$ are always pseudorandom—even if the plaintext $(x_1, x_2, \dots, x_s)$ has been carefully chosen by an attacker.

Although the security of our AONT is provable, the overall security of the proposed mode is only of a heuristic nature. Both step one and step three of our scheme are based on a block cipher whose expected behavior can hardly be proven. Nevertheless, we would like to stress the fact that (in contrast to separable encryption modi or the non-separable mode of Rivest) in our mode *an adversary does not even know the ciphertext.* Hence he cannot directly attack the cipher in a usual manner. For any of the possible values K' there exists a unique pseudorandom combination of ciphertexts $(y_0, y_1, \dots, y_s)$. Without the knowledge of K any of the 2^n possible sequences for each y_i looks equally bad or good, so the adversary cannot distinguish the right one. Moreover, because of the one-way property of the function f_w it is not enough to perform an exhaustive search for the value K'. One must exhaustively look directly for the secret key K, and the resulting attack is slowed down by factor w because of the w-slow f_w. Also a differential- or linear-like cryptanalysis becomes extremely hard, because after encrypting two different plaintexts one cannot tell in what extent the resulting output difference has been caused by the difference of the plaintexts or by the difference of IV, which is chosen randomly for every encryption.

In comparison with Rivest's scheme the security gain of our scheme is given by the constant w, which is freely variable and does not depend on the message length. This makes it possible to achieve a high protection improvement even for very short messages.

Certainly, our construction is not secure in B o y k o's semantic security model [3] as one can obtain, e.g., a sum of two neighbor ciphertext blocks. However, like the authors of [1], [2], and [4], we choose the block-wise notion of AON security for efficiency reasons.

Finally, we would like to stress the difference between our mode and the AONT setups used for encryption speedup or for remotely keyed encryption (see, e.g., Sec. 1 of [3]). In those setups one first applies a randomizing AONT, and then encrypts one of the output blocks. Our first two steps: the CBC encryption, and the non-randomizing AONT cannot be together considered as one randomizing AONT because of the secret key K involved. (By definition an AONT is an *unkeyed* transform.) It follows that our setup is principally different from the mentioned ones.

3.1 Non-expanding variant

In some applications the expanding property of an encryption mode might be considered as a serious disadvantage. In this case one can use a modified version of our mode in which a fixed IV is used instead of a random one. I.e., one can define, e.g., IV $= e_K(0)$ and compute $y_1 = e_K\big(e_K(0) \oplus x_1\big)$ instead of $y_1 = e_K(\mathrm{IV} \oplus x_1)$. In this setup no additional values y_0 (and consequently z_0) need to be transmitted or stored, and the resulting mode is *non-expanding*. However, one should be aware of two possible attacks against this simplified scheme.

First of all, a known-plaintext brute-force attack with complexity only $\mathcal{O}(2^k)$ (instead of $\mathcal{O}(2^{k+w})$) can be mounted in this case. Suppose the attacker knows one correct plaintext-ciphertext pair (x_1, z_1). Then he can try all possible key values K for computing $y_1' = e_K\big(e_K(0) \oplus x_1\big)$ and $z_1' = y_1' \boxplus e_K(0)$, until he finds the correct key which leads to $z_1' = z_1$.

In addition to the attack above, the attacker can perform a differential analysis (a chosen plaintext attack) even if he does not know the exact ciphertext. Note that by encrypting two messages $(x_1, x_2, \ldots, x_s)$ and $(x_1', x_2, \ldots, x_s)$ we obtain two output blocks $z_1 = e_K(0) + e_K\big(e_K(0) \oplus x_1\big)$ and $z_1' = e_K(0) + e_K\big(e_K(0) \oplus x_1'\big)$, and because $+$ is normally identical with $\oplus$, we are able to obtain a corresponding pair of input-output differences $\Delta_{\mathrm{in}} = x_1 \oplus x_1'$, and $\Delta_{\mathrm{out}} = z_1 \oplus z_1'$. This pair of differences can be used for a differential cryptanalysis. Nevertheless, we would like to stress that this second kind of attack is slowed down by factor w in comparison with a separable mode. So unless the underlying cipher is extremely vulnerable to a differential analysis, this idea is rather hypothetical.

4. Efficiency

In the Rivest's non-separable mode the slowdown of a brute-force attack is achieved by a straightforward combination of AONT and encryption. The AONT itself is the element which is slowing-down the adversary. In our mode the AONT still plays an important role, but the actual attack slowdown is achieved by the function f_w. This additional element might be, at the first view, considered as a kind of "artificial". Nevertheless, the simplicity of our AONT is more than a compensation for these additional operations.

In Table 1 we present the number of operations which have to be executed by variable message lengths s and security gain factors w.

TABLE 1. Operations to be executed.

s	w	Rivest's mode	Our mode		
10	10	$30 \times e_K$	$20 \times e_K$	$10\times$ add	$1\times$ mul
100	100	$300 \times e_K$	$200 \times e_K$	$100\times$ add	$1\times$ mul
1000	1000	$3000 \times e_K$	$2000 \times e_K$	$1000\times$ add	$1\times$ mul
10	10000	$30000 \times e_K$	$10010 \times e_K$	$10\times$ add	$1\times$ mull
100	10000	$30000 \times e_K$	$10100 \times e_K$	$100\times$ add	$1\times$ mull
1000	10000	$30000 \times e_K$	$11000 \times e_K$	$1000\times$ add	$1\times$ mull

In the first part we observe the case when $w = s$, i.e., when the desired security gain is equal to the number of message blocks. One can see that Rivest's construction needs to execute $3 \times s$ encryptions, while our construction performs $2 \times s$ encryptions, s additions, and one multiplication. As the field operations $+$ and $\cdot$ are significantly faster[1] than e_K, our overall processing time will be shorter—especially for long messages. In the case $w > s$, shown in the second part of the table, Rivest's mode must pad the message to a total length of w blocks, and execute 3 encryptions for each of them. Thus, the number of encryptions will be $3 \times w$. In our case only $s + w$ encryptions need to be performed. The additional s additions and one multiplication become negligible when $w \gg s$. All in all, our mode requires significantly less computing time than the mode by Rivest.

Finally, we would like to mention one implementational issue. The key setup procedure of some block ciphers is a very complex task, and it should be taken into account when implementing f_w. We denote the time needed for a key setup by t_0 and the time needed for one encryption by t_1. A recurrent computation of f_w (as proposed at the end of Section 2) invokes w key setups and w encryptions,

[1] Particularly the addition is extremely fast. For instance, when $q = 2^k$, the operation $+$ is in fact a simple XOR which requires only one processor cycle. The multiplication is noticeably slower but still faster than a usual encryption operation.

so its total computation time is $w \cdot (t_0 + t_1)$, not just $w \cdot t_1$. Consequently, in order to make the particular f_w w-slow (i.e., consuming $w \cdot t_1$ time units) it is enough to execute only $w' = w \cdot \frac{t_1}{(t_0+t_1)}$ iterations. Nevertheless, as the relation between t_0 and t_1 can vary from platform to platform, one should better be conservative when estimating w'.

4.1 Experimental results

In addition to the theoretical efficiency considerations we performed some practical measurements. The execution time of f_w is, in fact, given by its definition. On the other hand, the efficiency of ϕ depends on many implementation- and platform-dependent factors, e.g., memory speed, cache sizes, particular representation of the field elements, and others. Moreover, the relative overhead of a non-separable mode in comparison with a regular CBC mode depends significantly on the particular cipher used. For instance, our AONT implementation operating on a memory buffer (throughput 26 megabytes per second) consumed about 54 % of the time needed for an AES encryption, 18 % when compared to IDEA, and only 8 % when compared to DES. These ratios were obtained from a C++ implementation on a PC with 350 MHz Pentium II Processor and 100 MHz EDO RAM. It follows that the resulting non-separable mode composed from AES, AONT, and f_w, using security factor $w = s$, would consume about 2.54 times more time than a simple AES encryption (2.54 = 1.0 for the encryption + 0.54 for AONT + 1.0 for f_w). The slowdown factor for the same mode based on DES would be only 2.08.

When working with files on a hard disk, the relative time consumption of AONT in comparison with AES varies between 38 and 130 %.This variance depends significantly on the file length and is caused by disk caching. The worst relative AONT speed is measured for extremely long files (several tens of megabytes) when the caching techniques do not help anymore. The fact that our mode *does not* need to create very long messages to achieve a high security gain can be considered as an obvious advantage in this context. Furthermore, when working with hard disk, the time consumption caused by the evaluation of f_w becomes negligible in comparison with AONT or encryption. f_w itself does not namely need any hard disk operations. As a consequence, our composed non-separable mode working on hard disk consumes even in the worst case only about 2.3 times more time than a simple CBC encryption. For shorter files—up to few megabytes—which can be cached effectively, the slowdown factor is only about 1.6.

5. Conclusion

We proposed a new encryption mode which is "non-separable" in the sense that one has to execute a specified number of encryption operations before one can obtain a single plaintext block. As a consequence, our mode slows down any brute force attack based on an exhaustive key-search by a predefined factor w — similarly as the non-separable mode of Rivest. However, in contrast to Rivest's mode, our security improvement factor is freely adjustable, independently of the message length s. Moreover, because we apply the AONT on a strongly randomized input, we can use a simpler linear AONT construction and so increase the processing speed. The security of our encryption mode is block-wise and computational.

REFERENCES

[1] RIVEST, R. L.: *All-or-nothing encryption and the package transform*, in: Fast Software Encryption '97, Lecture Notes in Comput. Sci., Vol. 1267, Springer-Verlag, Berlin, 1997, pp. 210–218, `http://theory.lcs.mit.edu/~rivest/fusion.ps`.

[2] STINSON, D. R.: *Something about all-or-nothing* (*transforms*), Des. Codes Cryptogr. **22** (2001), 133–138, `http://cacr.math.uwaterloo.ca/~dstinson/papers/AON.ps`.

[3] BOYKO, V.: *On the security properties of OAEP as an all-or-nothing transform*, in: CRYPTO '99, Lecture Notes in Comput. Sci., Vol. 1666, Springer Verlag, Berlin, 1999, pp. 503–518, `http://theory.lcs.mit.edu/~boyko/aont-oaep/paper.ps`.

[4] DESAI, A.: *The security of all-or-nothing encryption: Protecting against exhaustive key search*, in: CRYPTO '00, Lecture Notes in Comput. Sci., Vol. 1880, Springer-Verlag, Berlin, 2000, pp. 359–375, `http://www-cse.ucsd.edu/users/adesai/papers/publ.html`.

[5] SCHNEIER, B.: *Applied Cryptography: Protocols, Algorithms and Source Code in C*, Second edition, J. Wiley and Sons, New York, 1996.

Received September 23, 2004

Institute for Experimental Mathematics
University of Duisburg-Essen
Ellernstrasse 29
G–45326 Essen
GERMANY

E-mail: valer@exp-math.uni-essen.de
trung@exp-math.uni-essen.de