One-Way IND-CNA Key Setup - a Step Towards Provably Secure Symmetric Encryption

Bartosz Zoltak

http://www.vmpcfunction.com
bzoltak@vmpcfunction.com

Abstract. We analyse the consequences of the specific properties of the key-setup phase in symmetric encryption schemes for their security. We find that key-setup routines satisfying IND-CNA and one-wayness allow to construct schemes which are provably secure against key-recovery attacks. We propose a specific cryptosystem for which we show that the key-setup routine ensures a significant increase in the security of the scheme regardless of the possible attacks against the underlying cipher. The paper presents a proof, based on a set of assumptions, that the scheme remains secure even if a successful key-recovery attack against the underlying cipher is found.

Keywords: provable security, stream cipher, cryptanalysis

1 Introduction

There has been a lot of interest in proofs of security for cryptographic schemes in recent years. A commonly employed approach is reduction and basing the proof on an assumption or a set of assumptions. This approach allows to build proofs that a given scheme is provably secure under an assumption that given primitives have certain properties or - generally - assuming that certain facts are true. An example result obtained by this approach would be to show that a given mode of operation of a block cipher is secure (i.e. that it satisfies the adopted notion of security under the adopted adversary model) assuming that the underlying cipher is secure.

The only known encryption scheme which enjoys unconditionally provable security is the One Time Pad, which does not limit its security to the key-space but to the message-space and all possible plaintexts of given length are equally likely to be correct. However this method has a significant drawback in its practical applications because a secure channel for transmitting a message-unique key of the same length as the size of the message is required.

Modern symmetric encryption algorithms are much easier to apply in practice than the OTP but they do not offer provable security. None of the algorithms has also been proved to require a brute-force search of the key-space, even though no better method of breaking a number of them is publicly known yet.

The most popular area for constructing reduction-based conditional proofs of security are modes of operations of block ciphers, like the CBC encryption mode or the EAX authenticated encryption mode. The proofs of this kind take the security of the underlying primitive (here - the block cipher) as a fundamental assumption.

However there has been little work on the proofs of security on a lower level, i.e. on the proofs of the security of the ciphers. This paper makes an attempt to go in that direction but still it does not present a proof of security of a cipher as such. Instead it discusses the properties of the key-setup phase of symmetric encryption schemes and their consequences for the security of the scheme. Specifically we describe that a key-setup routine which is a one-way function of argument *key* and a message-unique parameter *nonce* and which satisfies the IND-CNA (indistinguishability under chosen nonce attack) property can ensure provable security against key-recovery attacks on the underling cipher.

In sections 3-6 we describe a specific scheme of symmetric encryption of j messages with the same key. The scheme can be proved to remain secure even if the underlying cipher is broken in a notion that the secret key used for encryption is recovered. The proof is possible after assuming that the key-setup function is IND-CNA and one-way and after accepting a set of technical assumption specific for this particular scheme.

2 Inverting a composition of functions

Let f and g be functions: $f\colon X \rightarrow Y$ and $g\colon Y \rightarrow Z$ and let $x \in X$, $y \in Y$, $z \in Z$.

Assume an adversary whose task is to find the value of x. Assume the adversary knows the value of z, where $z = g(f(x))$ and that she can easily find the value of y given $z = g(y)$.

In the described scenario the adversary is unable to accomplish her task unless she can recover x from $y = f(x)$.

In other words:

Theorem 1. *Even if the adversary can invert $g(y)$, to recover x from $g(f(x))$, she needs to invert $f(x)$.*

Proof. Assume the adversary cannot invert $f(x)$. Then the adversary is unable to recover x from $g(f(x))$, because, even if she knows the value of y, where $y = f(x)$, by definition she needs to invert f. $\square$

2.1 Implications for symmetric encryption schemes

A common notation for symmetric encryption schemes is $Ct = E(Pt, K)$, where Ct and Pt denote the ciphertext and plaintext data respectively, K denotes a secret key and E denotes an encryption algorithm. The security of such scheme can be formalised in very strict terms, like IND-CPA (indistinguishability under chosen plaintext attack), which gives the adversary a significantly easier task to accomplish than in the more classical, but also more demanding for the adversary, model - where she needs to recover the secret key K. We might informally agree that recovering K is a greater threat for the secrecy of the encrypted data than distinguishing two chosen plaintexts by observing their ciphertexts. This is not to state that schemes only failing to satisfy IND-CPA are secure, but that schemes with feasible key-recovery attacks are less secure.

In consequence a hypothetical encryption scheme with provable security against key-recovery attacks could be considered desirable.

Using an analogy to Theorem 1 we can modify the model of the symmetric encryption scheme into $Ct = E(Pt, f(K))$, where f is any one-way function with a codomain compatible with the input to E. Any attack recovering the key $f(K)$ would be no threat for the secrecy of K if f is one-way. The adversary - apart from recovering the secret key used for encryption - here $f(K)$ - would still need to invert f to find the secret key K.

This however is obviously redundant - the knowledge of $f(K)$ is sufficient for the attacker to decrypt new messages, as she can input $f(K)$ to the decryption oracle and the value of K is then irrelevant.

However a simple modification to the scheme would actually force the adversary to invert f before she could decrypt a new message. Assume that the scheme is defined as $Ct = E(Pt, g(K, N))$, where N is a nonce (N has different value for each encrypted message and does not require secrecy) and g is any function with a codomain compatible with the input to E such that: (a) recovering K from $g(K, N)$ is infeasible even for a known value of N and (b) g is IND-CNA (indistinguishable under chosen nonce attack, i.e. an adversary who chooses N_1 and N_2 and observes $G_1 = g(K, N_{1+R})$ and $G_2 = g(K, N_{2-R})$, where $Prob(R = 0) = Prob(R = 1) = 1/2$, is unable to tell whether $G_1 = g(K, N_1)$ or $G_1 = g(K, N_2)$ with probability higher than 1/2).

Even if the adversary is able to perform a key-recovery attack and find the value of $g(K, N_1)$, each new message will be encrypted with a pseudo-randomly different (as a result of the IND-CNA property) $g(K, N_2)$ and thus the adversary knowing the value of $g(K, N_1)$ will have no advantage in decrypting a new message unless she can find $g(K, N_2)$. However, given the IND-CNA property of g, the adversary would need to invert g and find K to be able to produce $g(K, N_2)$ and decrypt the new message.

The outlined approach obviously does not guarantee that there exist no other attacks against E and that the encryption scheme cannot be broken using a different approach. Let A denote a set of all other attacks. We do not define A since it is intended to comprise all algorithms breaking E, regardless of the definition of breaking, i.e. A represents any possible threat to the security of the encryption scheme E.

The observations discussed in this section are concluded in the following theorem:

Theorem 2. *If there exists a set of feasible attacks A and a feasible key-recovery attack (finding h(K,N)) against an encryption scheme $Ct = E(Pt, h(K, N))$, then for an IND-CNA and one-way function g, the encryption scheme $Ct = E(Pt, g(K, N))$ can only be broken with A.*

Theorem 2 states that a one-way and IND-CNA key-setup for an encryption scheme E delivers an additional level of resistance against key-recovery attacks - recovering the key (K) requires two conditions to be satisfied: an actual key recovery attack on E has to be performed and the key-setup routine $g(K, N)$ needs to be inverted.

In the further part of this paper a specific cryptosystem designed to satisfy Theorem 2 will be described and analysed.

3 General description of VMPC Cryptosystem

VMPC Cryptosystem is based on VMPC stream cipher and VMPC Key Scheduling Algorithm (KSA), which were proposed at FSE'04 in "VMPC One-Way Function and Stream Cipher".

VMPC Cryptosystem specifies a method of transmitting j messages encrypted with the same key K and with a message-unique nonce N_i by Party A to Party B. The KSA of the cryptosystem was designed to ensure the IND-CNA property and to be practically one-way.

The cryptosystem consists of two major components - the KSA, which specifies an algorithm of transforming a secret key K and a nonce N_i into a 256-element permutation P and a stream cipher, extending P into a keystream used to encrypt i-th message.

Section 5 presents a conditional proof that a feasible attack recovering the cipher's internal state (P) is no threat to the security of the cryptosystem.

4 VMPC Cryptosystem

Notation:

$Plaintext_i[m]$: m-th byte of i-th message
$Ciphertext_i[m]$: m-th byte of encrypted i-th message

P_i : 256-byte table storing a permutation
s : 8-bit variable
K : c-byte table storing the key; $16 \leq c \leq 64$
N_i : z-byte table storing the nonce; $16 \leq z \leq 64$
$+$: addition modulo 256

```
1. Parties A and B set up the key ($K$), which is known only to A and B
2. Party A transmits $j$ messages to Party B by following steps 3-8:

3. For i from 1 to $j$ execute steps 4-8:

4. Generate a nonce $N_i$ such that $N_i \neq N_d$ for any $d \neq i$
5. Input $K$ and $N_i$ to VMPC KSA by executing steps 5.1 - 5.4.2
   $[P_i = KSA(KSA(KSA(P^{(0)}, K), N_i), K)]$:

   5.1. $s = 0$; for $n$ from 0 to 255: $P^{(0)}[n] = n$; $P_i = P^{(0)}$
   5.2. for $m$ from 0 to 767: execute steps 5.2.1 - 5.2.2:
        5.2.1. $s = P_i[s + P_i[m \bmod 256] + K[m \bmod c]]$
        5.2.2. $x = P_i[m \bmod 256]$; $P_i[m \bmod 256] = P_i[s]$; $P_i[s] = x$
   5.3. for $m$ from 0 to 767: execute steps 5.3.1 - 5.3.2:
        5.3.1. $s = P_i[s + P_i[m \bmod 256] + N_i[m \bmod z]]$
        5.3.2. $x = P_i[m \bmod 256]$; $P_i[m \bmod 256] = P_i[s]$; $P_i[s] = x$
   5.4. for $m$ from 0 to 767: execute steps 5.4.1 - 5.4.2:
        5.4.1. $s = P_i[s + P_i[m \bmod 256] + K[m \bmod c]]$
        5.4.2. $x = P_i[m \bmod 256]$; $P_i[m \bmod 256] = P_i[s]$; $P_i[s] = x$

6. Encrypt $i$-th message with VMPC stream cipher by executing steps 6.1-6.1.3:
   $[Ciphertext_i = Plaintext_i$ xor $SC(P_i)]$

   6.1. For $m$ from 0 to ($Length$ of $i$-th message)$-1$ execute steps 6.1.1 - 6.1.3:
        6.1.1. $s = P_i[s + P_i[m \bmod 256]]$
        6.1.2. $Ciphertext_i[m] = Plaintext_i[m]$ xor $P_i[P_i[P_i[s]] + 1]$
        6.1.3. $x = P_i[m \bmod 256]$; $P_i[m \bmod 256] = P_i[s]$; $P_i[s] = x$

7. Prepend $N_i$ to $Ciphertext_i$
8. Transmit $Ciphertext_i$ to Party B
```

5 Conditional proof of security of VMPC Cryptosystem

Note: "determining X" is assumed to state determining any information about the value of X.

Definition 1. *Breaking the cryptosystem: Determining $Plaintext_i$ given $Ciphertext_i$ and given $Ciphertext_d$ and $Plaintext_d$ for any values of $d \neq i$*

Definition 2. *Attack on the cipher: An algorithm recovering P_d from $SC(P_d)$, where $SC(P_d)$ = $Ciphertext_d$ xor $Plaintext_d$*

Observation 1. According to theoretical analyses and statistical tests described in [8] there is no known method of distinguishing $SC(P_i)$ from a truly random data stream.

Observation 2. According to [8] one phase of the KSA (768 iterations in steps 5.2 - 5.2.2 or 5.3 - 5.3.2 or 5.4 - 5.4.2.) provides an undistinguishable from random diffusion [1] of changes of one bit or byte of key of size up to 64 bytes onto the generated permutation P_x and onto output generated by the VMPC cipher ($SC(P_x)$ in steps 6.1 - 6.1.3.). The repetition of the 768 steps of the KSA in steps 5.4 - 5.4.2 additionally provides the diffusion effect with a significant safety margin. From these results we conjecture that the transformation $P_i = KSA(KSA(KSA(P^{(0)}, K), N_i), K)$, as specified in steps 5.1-5.4.2, provides the IND-CNA property, defined in Section 2.1.

Assumption 1. Let A be an adversary whose task is to find the value of $Plaintext_i$, where $Plaintext_i = Ciphertext_i$ xor $SC(P_i)$, and who has no information about the value of $SC(P_i)$. Let Adv_1 denote the probability that A finds the value of $Plaintext_i$. Then:

$$Adv_1 = 2^{-8\times c+1}$$

Justification. Following Assumption 1, the necessary information which could enable determining $Plaintext_i$ is the value of $SC(P_i)$. According to Observation 1, $SC(P_i)$ cannot be distinguished from random. As a result $Ciphertext_i = Plaintext_i$ xor $SC(P_i)$ is also undistinguishable from random, which thwarts the possibility of deducing any information about $Plaintext_i$ directly from $Ciphertext_i$.

Out of the remaining sources of information in the described cryptosystem only $SC(P_i)$ appears to be sufficiently related to $Plaintext_i$ to enable determining $Plaintext_i$.

[1] Relations between $P_1 = KSA(P^{(0)}, K_1)$ and $P_2 = KSA(P^{(0)}, K_2)$ and relations between $SC(P_1)$ and $SC(P_2)$, where K_1 differs from K_2 in one bit or one byte, are undistinguishable from relations expected between - respectively - two random permutations or two random data-streams

Assumption 2. Let A be an adversary whose task is to find the value of $SC(P_i)$, where $SC(P_i)$ is derived in steps 6.1-6.1.3, and who has no information about the value of P_i. Let Adv_2 denote the probability that A finds the value of $SC(P_i)$. Then:

$$Adv_2 = 2^{-8 \times c+1}$$

Justification. Following Assumption 2, the necessary information which could enable determining $SC(P_i)$ is the value of P_i. According to Observation 1, $SC(P_i)$ cannot be distinguished from random. As a result $Ciphertext_i = Plaintext_i$ xor $SC(P_i)$ is also undistinguishable from random, which thwarts the possibility of deducing any information about $SC(P_i)$ directly from $Ciphertext_i$.

According to Observation 2, relations between $SC(P_i)$ and $SC(P_d)$ and between P_i and P_d are undistinguishable from random, which thwarts the possibility of deducing any information about $SC(P_i)$ from $SC(P_d)$ for $d \neq i$.

Out of the remaining sources of information in the described cryptosystem only P_i appears to be sufficiently related to $SC(P_i)$ to enable determining $SC(P_i)$.

Assumption 3. Let A be an adversary whose task is to find the value of P_i, where $P_i = KSA(KSA(KSA(P^{(0)}, K), N_i), K)$, and who has no information about the value of K. Let Adv_3 denote the probability that A finds the value of P_i. Then:

$$Adv_3 = 2^{-8 \times c+1}$$

Justification. Following Assumption 3, the necessary information which could enable determining P_i, where $P_i = KSA(KSA(KSA(P^{(0)}, K), N_i), K)$, is the value of K. According to Observation 2 relations between $P_i^{(2)} = KSA(KSA(P^{(0)}, K), N_i)$ and $P_d^{(2)} = KSA(KSA(P^{(0)}, K), N_d)$ and, as a result of steps 5.4 - 5.4.2, relations between P_i and P_d are undistinguishable from random for $N_i \neq N_d$, which thwarts the possibility of determining P_i from P_d for $d \neq i$.

Out of the remaining sources of information in the described cryptosystem only K appears to be sufficiently related to P_i to enable determining P_i.

Assumption 4. Let A be an adversary whose task is to find the value of K and who has no information about the value of P_d, where $P_d = KSA(KSA(KSA(P^{(0)}, K), N_d), K)$ for any value of $d \in \{1, 2, \ldots, j\}$. Let Adv_4 denote the probability that A finds the value of K. Then:

$$Adv_4 = 2^{-8\times c+1}$$

Justification. Following Assumption 4, the necessary information which could enable determining K is the value of $P_d = KSA(KSA(KSA(P^{(0)}, K), N_d), K)$. In the described cryptosystem K is used nowhere else than for the computation of $P_d = KSA(KSA(KSA(P^{(0)}, K), N_d), K)$ for $d \in \{1, 2, \ldots, j\}$.

Theorem 3. *Even if an adversary performs a successful attack on the cipher and recovers P_d from $SC(P_d)$, to break the cryptosystem and determine $Plaintext_i$ for $i \neq d$, she needs to invert $P_d = KSA(KSA(KSA(P^{(0)}, K), N_d), K)$ and recover K, where N_d is a known parameter.*

Proof. Theorem 3 is true provided that assumptions 1-4 are true. Following Assumption 1, breaking the cryptosystem cannot be accomplished without determining $SC(P_i)$; following Assumption 2, $SC(P_i)$ cannot be determined without determining P_i; following Assumption 3, P_i cannot be determined without determining K; following Assumption 4, K cannot be determined without determining P_d for $d \neq i$. If P_d is known, which is assumed to be true as a result of a successful attack, then recovering [2] K from P_d is a necessary condition for breaking the cryptosystem. □

5.1 The problem of recovering K from P_d

We propose to assume that recovering any information about the value of K from P_d with probability higher than $2^{-8\times c}$ requires searching through the $2^{8\times c}$ possible values of K until one of the values conforms to
$P_d = KSA(KSA(KSA(P^{(0)}, K), N_d), K)$.

This assumption would be hard to prove because the transformation
$P_d = KSA(KSA(KSA(P^{(0)}, K), N_d), K)$ has a very complex algebraic structure. It performs 2304 operations, each constituting (a) an update of the s variable through combining a consecutive byte of K, the previous value of s and two elements of P_d and (b) a swap operation of a consecutive-modulo-256 and s-th element of P_d. As a result each element of P_i undergoes an average number of 18 changes in the course of the transformation. We believe that tracking these operations in the purpose of providing a proof of their uninvertibility is unreachable.

On the other hand the extent of the mixing the transformation performs on P_d indicates that the transformation might actually be infeasible to invert.

[2] The knowledge of all the bits of K is necessary in consequence of the diffusion effect described in Observation 2.

The transformation can be formulated as $P_d = KSA(P_d^{(2)}, K)$, where both $P_d^{(2)}$ and K are unknown values. $P_d^{(2)}$ is determined by the value of K and it denotes the middle-state of the P_d permutation after two 768-step phases of the transformation ($P_d^{(2)} = KSA(KSA(P^{(0)}, K), N_d)$).

Note that if a 2-phase KSA transformation was used in the cryptosystem ($P'_d = KSA(KSA(P^{(0)}, K), N_d)$), inverting it could be reduced to the problem of recovering $KSA(P^{(0)}, K)$ from $KSA(KSA(P^{(0)}, K), N_d)$. Recovering K would be unnecessary to break the cryptosystem because the known values $KSA(P^{(0)}, K)$ and N_i would allow to reconstruct $P'_i = KSA(KSA(P^{(0)}, K), N_i)$, generate the keystream $SC(P'_i)$ and recover $Plaintext_i$. This would be possible since given $P'_d = KSA(KSA(P^{(0)}, K), N_d)$ and the known value of N_d we could backtrack the operations performed by the KSA with the guess-work factor limited only to the value of s.

However had the value of N_d been unknown in the above situation, the backtracking process in each step would encounter an unknown value of $N_d[m \bmod z]$, where $m = 255, 254, ..., 255, ..., 255, ..., 0$ and this value would need to be guessed. This would result in $(z+1)$ guessed values after z steps, which is directly equivalent to searching through all the possible values of N_d and expecting success in half-way through.

As a result of the third phase of the KSA transformation in the actual cryptosystem, the guess-work required by the backtracking algorithm would be greater because both the value of K (an analogy to the unknown N_d for the 2-phase KSA scenario), and the value of the middle-state $P_d^{(2)} = KSA(KSA(P^{(0)}, K), N_d)$ would be unknown. Each step of the third phase of the KSA usually uses one byte of K and 3 elements of the $P_d^{(2)}$ permutation, all of which are unknown, which implies that 4 guesses per each step are required for the 3-phase KSA to proceed with the backtracking process.

From the above observations we conjecture Assumption 5:

Assumption 5. The most efficient method of recovering K from $P_d = KSA(KSA(KSA(P^{(0)}, K), N_d), K)$ is searching through the $2^{8 \times c}$ possible values of K until one of the values conforms to $P_d = KSA(P_d^{(2)}, K) = KSA(KSA(KSA(P^{(0)}, K), N_d), K)$, where P_d is assumed to be known as a result of a successful attack and N_d is a known parameter.

Provided that Assumption 5 is true, we can formulate a theorem:

Theorem 4. *Searching through the possible* $2^{8\times c}$ *values of* K *is a necessary condition for breaking the cryptosystem.*

Proof. Theorem 4 is a direct consequence of Theorem 3 and Assumption 5. □

6 Consequences of the proof for the cryptosystem

The key component of the described cryptosystem is the third phase of the Key Scheduling Algorithm (steps 5.4 - 5.4.2) without which the cryptosystem has a standard level of security, where determining a new plaintext is easily possible if a practical algorithm recovering P_d from $SC(P_d)$ is found (rather than the still believed optimal 2^{900}-operation algorithm described in [8])
[then $KSA(P^{(0)}, K)$ would be relatively easily recoverable from
$P_d = KSA(KSA(P^0, K),N_d)$, executing steps 5.3 - 5.3.2 for a known N_i would compute $P_i = KSA(KSA(P^0, K), N_i)$, which allows to reconstruct $SC(P_i)$ and decrypt $Ciphertext_i$].

A cryptosystem with the additional transformation of P_i in steps 5.4 - 5.4.2 remains secure even if a practical algorithm recovering P_d from $SC(P_d)$ is found. This, even if based on the assumptions described in sections 5 and 5.1, is an improvement of the security level which is ensured by other symmetric encryption schemes, which are considered broken if their internal state (here P_d) can be recovered.

In the described cryptosystem it is assumed that an adversary can recover the cipher's inernal state (P_d), but still the cryptosystem can be proved secure (under the described assumptions) because the internal state (P_i) used to encrypt each message is pseudorandomly different (following the IND-CNA property) and the KSA transformation is practically one-way.

In consequence the described scheme offers what might be called a two-layer level of security: recovering the internal state from the cipher's output (recovering P_d from $SC(P_d)$) is still believed to require an average effort of about 2^{900} operations, but even if this is overcome, breaking the cryptosystem would additionally require a feasible algorithm recovering K from
$KSA(KSA(KSA(P^{(0)}, K), N_d), K)$ or abolishing one of the 4 technical assumptions described in section 5.

The technical assumptions for the proof were intended to be as elementary and easily acceptable as possible. We believe it would be unrealistically optimistic to hope to prove the security of a cryptosystem without basing on assumptions. This compromise appears to be a necessary condition for the proof to be possible and, although the proof might be considered not a significant theoretical result, it appears not to diminish its practical value.

7 Conclusions

In this paper we analysed the role of the properties of the key-setup phase of symmetric encryption schemes and their consequences for the security of the scheme against attacks aimed at the recovery of the secret key. We described that designing one-way key-setup routines with IND-CNA property for nonce-based cryptosystems can provide a significant improvement in the security ensured by the scheme against key-recovery attacks.

We proposed a specific cryptosystem aimed at satisfying the criteria of the discussed theoretical observations to illustrate the consequences of a properly designed key-setup algorithm for the improvement of the security of the encryption scheme.

We made an attempt to construct a proof of security of the cryptosystem (Theorem 4) based on both the properties of the key-setup routine and on a set of technical assumptions, which appeared indispensable to allow the proof to be completed. Although the assumptions relate to many components of the cryptosystem, they were meant to be as elementary and justified as possible, and this way not to significantly diminish the practical value of the proof. Most of all the presented conditional proof was intended to illustrate that the key-setup routine meeting certain criteria allows to construct cryptosystems with significant security advantages, which brings them a step closer to the level of provable security.

References

1. Mihir Bellare: Practice-Oriented Provable Security, Proceedings of First International Workshop on Information Security, LNCS vol. 1396, Springer-Verlag 1999.
2. Jaechul Sung, Sangjin Lee, Jongin Lim, Seokhie Hong, Sangjoon Park: Provable Security for the Skipjack-like Structure against Differential Cryptanalysis and Linear Cryptanalysis, Proc. of ASIACRYPT 2000, LNCS vol. 1976, Springer-Verlag 2000.
3. Ju-Sung Kang, Sang-Uk Shin, Dowon Hong, Okyeon Yi: Provable Security of KASUMI and 3GPP Encryption Mode f8, Proceedings of ASIACRYPT 2001, LNCS vol. 2248, Springer-Verlag 2001.
4. Ross Anderson, Eli Biham: The practical and Provably Secure Block Ciphers: BEAR and LION, Proceedings of Fast Software Encryption 1996, LNCS vol. 1039, Springer-Verlag 1996.
5. Mihir Bellare, Anand Desai, Eron Jokipii, Phillip Rogaway: A Concrete Security Treatment of Symmetric Encryption, Proceedings of the 38th Annual Symposium on Foundations of Computer Science (FOCS '97), IEEE 1997.
6. Mihir Bellare, Philip Rogaway, David Wagner: The EAX Mode of Operation Pre-proceedings of Fast Software Encryption 2004, pages 367-384.
7. Tadayoshi Kohno, John Viega, Doug Whiting: CWC: A High-Performance Conventional Authenticated Encryption Mode, Pre-proceedings of Fast Software Encryption 2004, pages 385-402.
8. Bartosz Zoltak: VMPC One-Way Function and Stream Cipher, Pre-proceedings of Fast Software Encryption 2004, pages 190-204.