Blockwise-Adaptive Attackers
Revisiting the (in)security of some provably secure Encryption Modes: CBC, GEM, IACBC

Antoine Joux, Gwenaëlle Martinet, and Frédéric Valette

DCSSI Crypto Lab
18, rue du Docteur Zamenhof
92131 Issy-les-Moulineaux
Antoine.Joux@m4x.fr
Gwenaelle.Martinet@ens.fr
fred.valette@wanadoo.fr

Abstract. In this paper, we show that the natural and most common way of implementing modes of operation for cryptographic primitives often leads to insecure implementations. We illustrate this problem by attacking several modes of operation that were proved to be semantically secure against either chosen plaintext or chosen ciphertext attacks.
The problem stems from the simple following fact: in the definition and proofs of semantic security, messages are considered as atomic objects that cannot be split; however, in most practical implementations, messages are subdivided into smaller chunks than can be easily manipulated. Depending on the implementation, each chunk may consist of one or several blocks of the underlying primitive. The key point here is that upon reception of a processed chunk, the attacker can now adapt his choice for the next chunk. Since the possibility of adapting within a single message is not taken into account in the current security models, this leaves room for unexpected attacks.
We illustrate this new paradigm by attacking three symmetric and hybrid encryption schemes based on the chaining mode in spite of their security proofs.

1 Introduction

Currently, the strongest definition of security for an encryption scheme captures the idea that an attacker can adapt his queries according to the previously received answers. A scheme is said to be secure, if no attacker is able to distinguish between different scenarios. These definitions exist in several flavors, depending on the allowed scenarios, e.g. Find-Then-Guess (FTG) security, Left-or-Right (LOR) security, Real-or-Random (ROR) security ([2]). Moreover, the attacker can be given access to an encryption oracle only, when considering chosen-plaintext attacks (CPA), or to an encryption/decryption oracle when considering chosen-ciphertext attacks (CCA). The case of chosen ciphertext secure modes of operation has been specially studied in [11].

However, all these definitions consider messages as atomic objects that cannot be split into smaller pieces. While very convenient from a theoretical point of view, this approach does not really model the reality of many cryptographic implementations. Indeed, in real-life implementation, encryption is usually performed "on the fly", *i.e.* ciphertext chunks are computed and sent as soon as possible. The potential attacks induced by such implementations have already been taken into account in some cryptographic constructions, such as in [7] (signed digital streams) and in [6] (pseudorandom number generators). However, we are not aware of any work that takes advantage of this to attack practical implementations of previously existing schemes. The first example that comes to mind is the case of encryption with a smart card. Usually, the host computer sends blocks of plaintext one at a time to the smart card and immediately receives the corresponding ciphertext block. Thus an hostile host can adapt his next plaintext block as a function of the previously received ciphertext block. Even when the encryption is performed by a general purpose computer, messages are divided into smaller chunks. For example, in SSH ([14]), the plaintext to be encrypted is stored in a buffer of finite size. Whenever the buffer is full[1], it is encrypted and sent. Moreover, as described in [14], "initialization vectors should be passed from the end of one packet to the beginning of the next packet". As a consequence, even though attackers cannot be adaptive within a buffer, they can adapt from one buffer to the next, within a single message. Finally, even if several blocks are stored in the cryptographic component and if buffers are longer than one block, the attacker can force a dependency between the last block of a buffer and the first block of the next one.

In the rest of paper, we show how to take advantage of this extra degree of freedom to attack some modes of operations that were previously thought (and proven) secure. These cryptanalysis are presented in the Find-Then-Guess model, described in appendix A. For the sake of simplicity, we will allow the attacker to be adaptive from one block to the next within a single message. This mimics the behavior of smart card implementations. Throughout the paper, this kind of attacker is said to be blockwise-adaptive.

The first and simplest cryptanalysis we present is the attack on CBC mode of operation. The attacker adapts directly the plaintext block according to the previous ciphertext block. The proposed attack is very efficient, it uses a small constant number of queries to the encryption oracle and always succeeds.

The second attack is against an hybrid scheme, called GEM, that was proposed by Coron *et al.* in [5]. It is an academic attack of higher complexity in time and memory. However, the attack beats the bound of the security proof. The main idea is to feed the challenge oracle with message blocks until a collision appears on the ciphertexts blocks. Then with a single query to a decryption oracle the attacker can distinguish which message has been encrypted.

Finally we attack the IACBC encryption mode proposed by Jutla in [10] and proved secure by Halevi [8] in a slightly modified variant. Here the attack

[1] To avoid useless waits, the SSH layer can encrypt and send incomplete buffers. However, this detail is irrelevant at this point.

exploits some relations between the values used to mask the ciphertext blocks. As for CBC, the attack is very efficient since the attacker just needs to feed the encryption oracle with a constant number of queries and always succeeds. Furthermore, we show in appendix B that this weakness was already present in the initial proposal of Jutla.

2 Attack on the CBC mode of operation

The CBC (Cipher Block Chaining) mode of encryption security has been analyzed in [2]. It was proved to be secure in the LOR-CPA sense, assuming that the underlying block cipher is a family of PRP. The definition of this security notion is standard and can be found in [2]. It is also briefly described in appendix A. In this section, we briefly recall the CBC encryption mode and then we describe how it can be attacked when allowing the attacker to be adaptive from one block to the next within a single message.

Let E_K be a block cipher with secret key K and block-size n bits and let M be the (padded) message to encrypt. M is divided into ℓ n–bit blocks denoted by $(M[1], M[2], \ldots, M[\ell])$. A random n–bit initial value IV is generated by the encryption box. The CBC mode of encryption with random initial value is a stateless symmetric encryption scheme $CBC(\mathrm{E}_K)$. The ciphertext blocks $C[i]$ are computed as follows:

$$\begin{aligned} C[0] &= IV, \\ C[i] &= \mathrm{E}_K(C[i-1] \oplus M[i]) \end{aligned}$$

The transmitted ciphertext is $(C[0], C[1], \ldots, C[\ell])$.

The crux of the security proof of [2], is that since each message block is randomized by xoring it with a block cipher output, each new call to E_K is independent from the previous ones and no attacker can succeed unless a random collision occurs. However, if $C[i-1]$ is known when choosing $M[i]$, the independence is clearly lost and the proof fails.

It turns out that this can be illustrated by a very simple attack in the (blockwise) FTG-CPA sense. The attack proceeds as follows:

Step 1 The attacker chooses its FTG challenge. This challenge consists of two-blocks messages M_0 and M_1, such that $M_0[2] \neq M_1[2]$.
Step 2 The black-box computes the encryption of either M_0 and M_1, according to the value of a random bit b. It transmits $(C_b[0], C_b[1], C_b[2])$ to the attacker. The goal of the attacker is now to guess the value of b.
Step 3 The attacker starts the encryption of a test message M'. It first sends the first block $M'[1]$ chosen uniformly at random.
Step 4 The attacker receives the beginning of the encryption of M', namely $(C'[0], C'[1])$. It sends the second block $M'[2] = M_0[2] \oplus C_b[1] \oplus C'[1]$.
Step 5 The attacker receives $C'[2]$.
Step 6 If $C'[2] = C_b[2]$ the attacker guesses that $b = 0$, otherwise it guesses $b = 1$.

We claim that the guess of the attacker is always correct. Indeed, when $b = 0$ we can check that:

$$\begin{aligned} C'[2] &= \mathrm{E}_K(M_0[2] \oplus C_b[1] \oplus C'[1] \oplus C'[1]) \\ &= \mathrm{E}_K(M_0[2] \oplus C_b[1]) \\ &= C_b[2] \end{aligned}$$

Moreover, when $b = 1$ we can check that:

$$\begin{aligned} C'[2] &= \mathrm{E}_K(M_0[2] \oplus C_b[1] \oplus C'[1] \oplus C'[1]) \\ &= \mathrm{E}_K(M_0[2] \oplus C_b[1]) \\ &\neq \mathrm{E}_K(M_1[2] \oplus C_b[1]) \end{aligned}$$

and thus $C'[2] \neq C_b[2]$. As a consequence, the attacker can easily find which of the two challenge messages was encrypted.

One can remark that the proposed attack could be even more efficient. Indeed, if the attacker can be adaptive *during* the challenge phase itself (and not only after, as described above), the test message is no longer necessary and the adversary can guess the bit b by just seeing the challenge ciphertext.

A simple and efficient countermeasure to this attack could be considered. The encryption process $\mathcal{E}$ can delay the outputs by one block. That is, when receiving the kth plaintext block, $\mathcal{E}$ encrypts and stores it, and returns the $(k-1)$th block of the ciphertext. In this case, an adversary against this scheme cannot adapt each plaintext block according to the previous ciphertext block during the encryption process, and the above attack fails. This scheme will be called the Delayed Cipher-Block Chaining and will be denoted by DCBC.

Remark 1. The same cryptanalysis can also be mounted against the ABC encryption mode (Accumulated Block Chaining) proposed by Knudsen in [12]. However we do not explicitly describe the attack which is related to the proof by Bellare *et al.* in [1] that ABC mode of operation with public or secret initial value is not a secure OPRP. We just remark here that this attack is possible since each plaintext block is masked with the previous ciphertext block and with a value issued from a function h evaluated at the previous plaintext block. As the h function is not kept secret, the attacker can predict the mask values and adapt each message block accordingly.

Remark 2. In many cases, encrypted messages are also authenticated using a message authentication code (MAC). It is known from recent papers [3] that the right way of doing this is the Encrypt-Then-MAC paradigm. When encryption and authentication are correctly combined, the complete system was shown to be CCA secure in the current security model. However, it is easily remarked that adding authenticity does not prevent the above attack.

3 An hybrid example: the GEM schemes

Two chosen ciphertext secure asymmetric encryption schemes for messages of arbitrary length , GEM-1 and GEM-2, have recently been presented in [5]. In fact they are based on an hybrid construction using an asymmetric encryption scheme and a block cipher. The security proof is made in the random oracle model with a very weak assumption on the underlying block cipher: any fixed-length indistinguishable secure symmetric scheme can be used.

In this section we show how to cryptanalyze these schemes with help of our new kind of attacks. In order to simplify the analysis of the attack, we assume that the underlying symmetric encryption scheme is the XOR, as proposed in the original paper. We mount a chosen ciphertext attack in the sense of the indistinguishability of the encryption. This proposed attack is blockwise-adaptive in a stronger sense than the attack against CBC encryption. Indeed, in the case of GEM, the attacker needs to be adaptive during the challenge transmission phase. We will focus on the first scheme GEM–1, but the same attack can be mounted against the second, GEM–2.

3.1 Overview of GEM–1

Let us briefly describe the GEM–1 scheme according to [5]. The system makes use of several cryptographic primitives, a trapdoor one way function $\mathcal{E}_{pk}$ (such as RSA) and a weak symmetric encryption scheme E_K. In fact, using the XOR function is proposed by the authors. The scheme also makes use of a family of hash functions H_i and of an additional hash function F which are modeled as random oracles. For practical instantiations, it is proposed to use SHA-1 together with a counter, *i.e.* $H_i(.) = \text{SHA-1}(\ . \parallel i)$. The additional hash function F can be defined similarly using a special value for i, e.g. $F = H_0$.

Given the public key pk, one can encrypt a message M formed of n l–bit blocks, $(M[1], M[2], \ldots, M[n])$ by randomly choosing w and u and by computing the ciphertext $(T_1, C[1], C[2], \ldots, C[n], T_2)$ as follows:

$$\begin{aligned}
T_1 &= \mathcal{E}_{pk}(w, u) \\
k_1 &= H_1(w, T_1) \\
C[1] &= E_{k_1}(M[1]) \\
k_i &= H_i(k_{i-1}, M[i-1], w) \\
C[i] &= E_{k_i}(M[i]) \\
T_2 &= F(k_n, M[n], w)
\end{aligned}$$

This is summarized in figure 1.

3.2 Attack on GEM–1

The security of GEM–1 is proved in [5] in the random oracle model, assuming that $\mathcal{E}_{pk}$ is “reasonably” secure, even when E_K is quite weak (a simple XOR

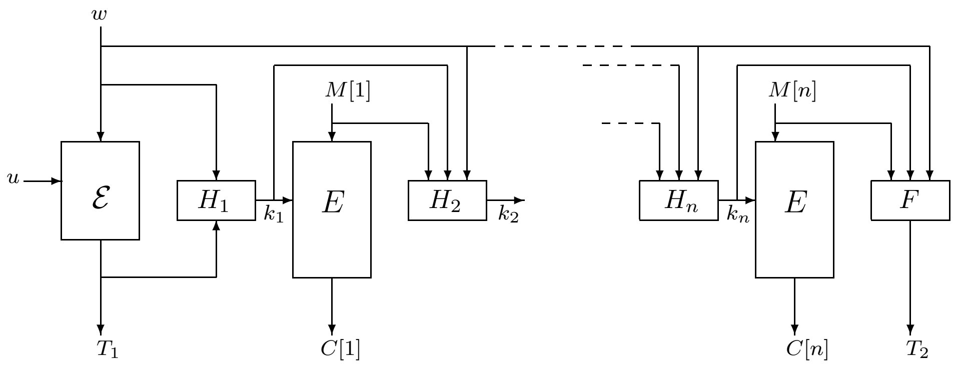

Fig. 1. The GEM–1 algorithm.

suffices). Without writing down the explicit security bound given for GEM–1, let us remark that the advantage of a CCA adversary in the usual security model is linear in the size of the processed data. As a consequence, square-root attacks are ruled out by the security proof.

In this section, we show that this is no longer the case when using a blockwise-adaptive attacker and give an explicit square-root attack using such an attacker. Note that the proposed attacker is blockwise-adaptive during the challenge phase itself.

The attacker needs to transmit a challenge of size $\mathcal{O}(2^{n/2})$ where n is the size in bits of the C_i values. In other words, this can be described as a square-root attack.

For the sake of simplicity, we assume that the XOR function is used as symmetric encryption. The important property of the XOR function for our purpose is that for a given pair consisting of one plaintext block and its related ciphertext block, the encryption key is uniquely determined. With a different block cipher algorithm, several keys could be possible. However, when the block size and the key size are both equal to n, the number of possible keys is always small. As a consequence, the proposed attack would still have a good probability of success with an ordinary block cipher.

After constructing its challenge message and getting the corresponding ciphertext, the attacker asks for the decryption of a different (but of course related) message. This decryption message tells him which of the two challenge messages was encrypted with probability 1.

The attack goes as follows:

Step 1 The attacker chooses and transmits the first block of each challenge message, $M_0[1]$ and $M_1[1]$, such that $M_0[1] \neq M_1[1]$. At this point in time, the attacker has not yet decided the length of the challenge messages.

Step 2 The encryption box computes the tag T_1, picks a bit b in $\{0, 1\}$, encrypts $M_b[1]$ and returns $(T_1, C_b[1])$.

Step 3 The attacker now sends the second block of each challenge message $M_0[2] = M_0[1]$ and $M_1[2] = M_1[1]$.
Step 4 The encryption box encrypts $M_b[2]$ and returns $C_b[2]$.
Step 5 The attacker continues to send the challenge messages one block at a time, with $M_0[i] = M_0[1]$ and $M_1[i] = M_1[1]$. It receives the encrypted blocks $C_b[i]$ and waits for a collision among these encrypted blocks.
Step 6 When a collision occurs, namely when the attacker receives a ciphertext block $C_b[i]$ such that there exists $j < i$ with $C_b[i] = C_b[j]$, the attacker tells the encryption box that the challenge messages are complete.
Step 7 The encryption box computes and returns the tag T_2.
Step 8 The attacker now requests the decryption of the truncated ciphertext $(T_1, C_b[1], \ldots, C_b[j], T_2)$. This decryption is either a truncation of M_0 or a truncation of M_1. The attacker guesses b accordingly.

In order to check that the attacker always succeeds, it suffices to verify the validity of the tag T_2 for the truncated message. For the original message, T_2 was computed as $F(k_i, M_b[i], w)$. When decrypting the truncated message, w is the same (since T_1 has not changed), and $M_b[j] = M_b[i]$ by choice of the challenge messages. Moreover, since $C_b[j] = C_b[i]$ thanks to the collision check performed by the attacker, we have $k_j = k_i$. As a consequence, $T_2 = F(k_j, M_b[j], w)$ is a valid tag for the truncated message and the truncated plaintext is indeed returned by the decryption box.

In order to determine the complexity of the attack, we must evaluate the expected length of challenge messages needed before a collision occurs. Thanks to the birthday paradox, since the keys and ciphertext blocks are coded on n bits, collisions are expected after $\mathcal{O}(2^{n/2})$ blocks. According to the security proof given in [5], no attack in the usual (non blockwise-adaptive) model can be that efficient.

4 Jutla's IACBC

In [10, 9], Jutla proposes two new encryption modes that provide confidentiality and integrity in a single pass. One of these modes, IACBC, is a CBC encryption of the plaintext where the encrypted blocks are hidden by xoring them with a sequence of masks $(S_0, \ldots, S_{\ell-1})$. For the scheme to be secure, these masks need to be pairwise independent within each encryption, however full pseudorandomness is not necessary. Furthermore, in [8] Halevi proposed a slight modification where he generates the mask values using a non cryptographic process. Halevi then proves the security of the modified scheme in the ROR-CCA sense. The proof is based on the pairwise independence of the masks. However in the sequel we use the fact that the masks are not truly independent to attack the scheme using a blockwise adaptive attacker.

4.1 Overview of IACBC

The IACBC mode works as follows: let E_K be a block cipher with block size n and key length k, along with secret key K_1. Let r be a random initial vector

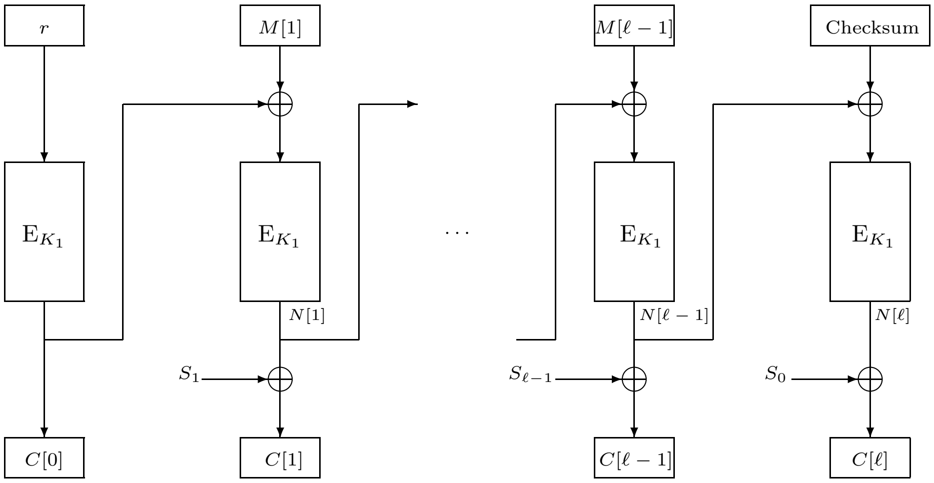

Fig. 2. The IACBC encryption mode.

of n bits used to generate ℓ mask values $S_0, S_1, \ldots, S_{\ell-1}$, where $\ell - 1$ is the size in blocks of the message to encrypt. In Jutla's paper [10], these masks are generated from $t = \lceil \log(\ell + 1) \rceil$ random and independent vectors W_i, computed from $r+1, \ldots, r+t$ using the block cipher with another secret key K_0. To speed up the generation of the masks, Gray codes are used. With this generation technique the masks are pairwise independent within a single encryption. Moreover the sequences of masks are independent between encryptions. In [8] Halevi proves that second property is not necessary to prove the security of the scheme. Then he proposes a new method to generate the masks values: let r a random initial vector of n bits, and M a secret random boolean matrix of dimension $n \times (\log \bar{L} + 1 + n)$, where $\bar{L}$ is an upper bound on the ciphertext length. For $j = 1$ to $\ell - 1$ we have $S_j = M \cdot (< 2j >, < r >)$, where $(< 2j >, < r >)$ is the boolean vector of length $\log \bar{L} + 1 + n$ composed with the binary representation of $2j$ on $\log \bar{L} + 1$ bits and the binary representation of r on n bits. Furthermore $S_0 = M \cdot (< 2L + 1 >, < r >)$, where L is the ciphertext length.

Then the ciphertext is generated as follows: the message is divided into $\ell - 1$ blocks $M[1], \ldots, M[\ell - 1]$, of n bits each. The ciphertext is defined by:

$C[0] = \mathrm{E}_{K_1}(r)$
$N[0] = C[0]$
`for` $i = 1$ `to` $\ell - 1$ `do`
  $N[i] = \mathrm{E}_{K_1}(M[i] \oplus N[i-1])$
  $C[i] = N[i] \oplus S_i$
`end for`
$C[\ell] = \mathrm{E}_{K_1}(\texttt{checksum} \oplus N[l-1]) \oplus S_0$, where $\texttt{checksum} = \bigoplus_{i=1}^{l-1} M[i]$.

This is summarized in figure 2.

To decrypt a ciphertext C, the receiver parses it into $\ell + 1$ blocks denoted by $(C[0], C[1], \ldots, C[\ell])$ and computes $r = \mathrm{D}_{K_1}(C[0])$. He can then recover the mask values $(S_0, \ldots, S_{\ell-1})$ with the help of the secret boolean matrix M. Each plaintext block $M[i]$ is computed as $M[i] = \mathrm{D}_{K_1}(C[i] \oplus S_i) \oplus C[i-1] \oplus S_{i-1}$. The message integrity is verified by checking the correctness of the `Checksum`.

4.2 Blockwise adaptive cryptanalysis

In this section we exhibit a cryptanalysis of the scheme, in the blockwise adaptive adversarial model. The main idea is to used deterministic relations verified by the masks. Indeed, even though the values used for different blocks are pairwise independent, by construction they satisfy some relations. For every set of masks $S = (S_0, S_1, \ldots, S_{\ell-1})$ and every pair of indices (i, j), $S_i \oplus S_j$ is a constant. To prove this claim we have to look at the mask generation. We have:

$$S_i = M \times (< 2i >, < r >) \text{ and } S_j = M \times (< 2j >, < r >)$$

Thus we get:

$$\begin{aligned} S_i \oplus S_j &= M \times (< 2i > \oplus < 2j >, < r > \oplus < r >) \\ &= M \times (< 2i > \oplus < 2j >, < 0 >) \end{aligned}$$

Then the vector $S_i \oplus S_j$ is independent of r and only depends on some columns of the secret matrix M. Thus, for every set of masks and every pair of indices, $S_i \oplus S_j$ is constant. In the attack we will use this fact for $S_1 \oplus S_2$.

The proposed blockwise adaptive attacker is adaptive during the encryption query but not during the challenge phase itself. However the encryption box has to send the initial ciphertext block $C[0]$ before it receives the first plaintext block.

Here is the scenario of the attack:

Step 1 The attacker chooses at random two messages of two blocks M_0 and M_1 at random and such that $M_0[1] = M_1[1]$ and $M_0[2] \neq M_1[2]$.

Step 2 The challenge box generates the masks values (S_0, S_1, S_2) from a random initial value r. It then picks at random a bit b, encrypts r and M_b under the secret key and transmits $C_b[0] \parallel C_b[1] \parallel C_b[2] \parallel C_b[3]$. The aim of the attacker is to guess the bit b.

Step 3 The attacker now queries the encryption box for one message of two blocks. It first receives $C'[0]$ and sends $M[1] = C'[0] \oplus M_0[1] \oplus C_b[0]$.

Step 4 After receiving $C'[1]$ the attacker outputs $M[2] = M_0[2]$. Then it receives $C'[2]$ and ends the query. The encryption box finally outputs $C'[3]$.

Step 5 if the equality $C_b[1] \oplus C_b[2] = C'[1] \oplus C'[2]$ holds, the attacker guesses the bit $b' = 0$, else he guesses $b' = 1$.

We claim that the attacker always guesses correctly the bit b. Indeed, suppose that message M_0 has been encrypted, meaning that $b = 0$. Then we get:

$$\begin{aligned} C_b[1] \oplus C_b[2] = {} & \mathrm{E}_K(M_0[1] \oplus C_b[0]) \oplus S_1 \\ & \oplus \mathrm{E}_K(M_0[2] \oplus \mathrm{E}_K(M_0[1] \oplus C_b[0])) \oplus S_2 \end{aligned}$$

Furthermore, we have:

$$\begin{aligned}
C'[1] \oplus C'[2] &= \mathrm{E}_K(M[1] \oplus C'[0]) \oplus S'_1 \\
&\quad \oplus \mathrm{E}_K(M[2] \oplus \mathrm{E}_K(M[1] \oplus C'[0]) \oplus S'_2 \\
&= \mathrm{E}_K(C'[0] \oplus M_0[1] \oplus C_b[0] \oplus C'[0]) \oplus S'_1 \\
&\quad \oplus \mathrm{E}_K(M[2] \oplus \mathrm{E}_K(C'[0] \oplus M_0[1] \oplus C_b[0] \oplus C'[0]) \oplus S'_2 \\
&= \mathrm{E}_K(M_0[1] \oplus C_b[0]) \oplus S'_1 \\
&\quad \oplus \mathrm{E}_K(M_0[2] \oplus \mathrm{E}_K(M_0[1] \oplus C_b[0])) \oplus S'_2
\end{aligned}$$

Now, we have proved above that $S_1 \oplus S_2 = S'_1 \oplus S'_2$. Consequently, if $b = 0$, we always have $C_b[1] \oplus C_b[2] = C[1] \oplus C[2]$.

Moreover, if $b = 1$ this equality never holds. Indeed, challenge messages M_0 and M_1 have been chosen such that $M_0[1] = M_1[1]$ and $M_0[2] \neq M_1[2]$, and the test message is such that $M[1] = C'[0] \oplus M_0[1] \oplus C_b[0]$. Then it is easy to check that in this case $C_b[1] \oplus C_b[2]$ never equals $C'[1] \oplus C'[2]$. Indeed, we have:

$$\begin{gathered}
S_1 \oplus S_2 = S'_1 \oplus S'_2 \\
M_1[1] \oplus C_b[0] = M[1] \oplus C'[0] \\
M_1[2] \oplus \mathrm{E}_K(M_1[1] \oplus C_b[0])) \neq M[2] \oplus \mathrm{E}_K(M[1] \oplus C'[0]))
\end{gathered}$$

and as a consequence $C'[1] \oplus C'[2] \neq C_b[1] \oplus C_b[2]$. Thus the attacker's guess of b is always correct.

The crucial step in this attack is the encryption query made by the adversary and the way in which the oracle returns the ciphertext blocks. Indeed, if the initial value is not sent before the beginning of the encryption, the adversary cannot adapt the next plaintext blocks and the attack fails. Thus, if correctly implemented, IACBC encryption scheme is not subject to such an attack.

Remark 3. Note that the initial IACBC scheme proposed by Jutla in [10] can be attacked in a similar way. Indeed, even when sequences of masks are independent between encryptions, it is however possible to find non trivial relations within a single encryption. This property can be used to cryptanalyze the scheme in the blockwise adaptive adversarial model. See appendix B for more details.

5 Conclusion

In this paper, we proposed a new class of attacks against modes of operation. These attacks, called blockwise adaptive, take advantage of the properties of most practical implementations to allow cryptanalysis of some modes that were previously thought (and proven) secure. Some other modes of operation do not seem to be vulnerable to such attacks, especially when there is no chaining (as in OCB, [13]), or when secret masks are used to randomized inputs and outputs of the block cipher (as XCBC, [4], and HPCBC, [1]). Furthermore, although the impact of this attack on the CBC is huge, this can be simply avoided by using the Delayed CBC (DCBC) that consists in delaying the outputs by one block.

We believe that dealing with blockwise adaptive attacks is the next step towards secure implementations of cryptographic modes of operation.

References

1. M. Bellare, A. Boldyreva, L. Knudsen, and C. Namprempre. On-Line Ciphers and the Hash-CBC Construction. In J. Kilian, editor, *Advances in Cryptology – Crypto'01*, volume 2139 of *Lecture Notes in Computer Science*, pages 292 – 309. Springer-Verlag, Berlin, 2001.
2. M. Bellare, A. Desai, E. Jokipii, and P. Rogaway. A Concrete Security Treatment of Symmetric Encryption. In *Proceedings of the 38th Symposium of Fundations of Computer Science*. IEEE, 1997.
3. M. Bellare and C. Namprempre. Authenticated Encryption: Relations among notions and analysis of the generic composition paradigm. In T. Okamoto, editor, *Advances in Cryptology – Asiacrypt'00*, volume 1976 of *Lecture Notes in Computer Science*. Springer-Verlag, Berlin, 2000.
4. V.D. Gligor and P. Donescu. Fast Encryption and Authentication: XCBC and XECB Authentication Modes. In *Fast Software Encryption*, Lecture Notes in Computer Science. Springer-Verlag, Berlin, 2001.
5. J.S Coron, H. Handshuh, M. Joye, P. Paillier, D. Pointcheval, and C. Tymen. Real-life Chosen-Ciphertext Secure Encryption of Arbitrary-Length Messages. In D. Naccache, editor, *PKC'2002*, volume 2274 of *Lecture Notes in Computer Science*, pages 17 – 33. Springer-Verlag, Berlin, 2002.
6. A. Desai, A. Hevia, and Y.L Yin. A Practice-Oriented Treatment of Pseudorandom Number Generators. In L. Knudsen, editor, *Advances in Cryptology – Eurocrypt 2002*, volume 2332 of *Lecture Notes in Computer Science*. Springer-Verlag, Berlin, 2002.
7. R. Gennaro and P. Rohatgi. How to Sign Digital Streams. In Burt Kaliski, editor, *Advances in Cryptology – Crypto'97*, volume 1294 of *Lecture Notes in Computer Science*, pages 180 – 197. Springer-Verlag, Berlin, 1997.
8. S. Halevi. An Observation regarding Jutla's modes of operation. Crytology ePrint archive, Report 2001/015, available at `http://eprint.iacr.org`.
9. C. Jutla. Encryption modes with almost free message integrity. Cryptology ePrint archive, Report 2000/039, available at `http://eprint.iacr.org`.
10. C. Jutla. Encryption modes with almost free message integrity. In B. Pfitzmann, editor, *Advances in Cryptology – Eurocrypt'01*, volume 2045 of *Lecture Notes in Computer Science*. Springer-Verlag, Berlin, 2001.
11. J. Katz and M. Yung. Unforgeable Encryption and Chosen Ciphertext Secure Modes of Operation. In Bruce Schneier, editor, *Fast Software Encryption*, volume 1978 of *Lectures Notes in Computer Science*. Springer-Verlag Berlin, 2000.
12. L. Knudsen. Block chaining modes of operation. Technical report, Department of Informatics, University of Bergen, 2000.
13. P. Rogaway, M. Bellare, J. Black, and T. Krovetz. OCB: A Block-Cipher Mode of Operation for Efficient Authenticated Encryption. In *Eighth ACM conference on Computer and Communications Security*. ACM Press, 2001.
14. T. Ylonen, T. Kivinen, M. Saarinen, T. Rinne, and S. Lehtinen. SSH Transport Layer Protocol, Network Working Group. January 2002. Internet-Draft available at `http://www.ietf.org/internet-drafts/draft-ietf-secsh-transport-12.txt`.

A Security notions

In the standard model, privacy of an encryption scheme is viewed as ciphertext indistinguishability. In [2] the authors have defined different security notions and proved that the strongest one is the LOR ("Left or Right). However, we focus here on the Find-Then-Guess (FTG) model. We can modelize this notion through a "Find-Then-Guess" game. In this setting the adversary is first given access to an encryption oracle $\mathcal{E}$ that he can feed with plaintexts of his choice. At the end of the first phase (the "Find" phase) the adversary returns two plaintexts M_0 and M_1 of equal length. The encryption oracle flips a bit b, encrypts M_b and returns the challenge ciphertext C_b. The adversary's goal is to guess with non negligible advantage the bit b. In the "Guess" phase, he is again given access to the encryption oracle, he can feed with plaintexts of his choice At the end of the game, the adversary returns a bit b' representing his guess. This attack is called a *Chosen Plaintext Attack* (CPA). However the adversary can also performed *Chosen Ciphertext Attacks* (CCA). In this setting, he also has access to a decryption oracle he can feed with queries of his choice, except with the challenge ciphertext C_b itself.

A symmetric encryption scheme is said to be FTG-CPA secure (respectively FTG-CCA secure), if no polynomial time adversary can guess the bit b in the respective games, with non negligible advantage.

B Cryptanalysis of the original Jutla's IACBC

In the original Jutla's proposal in [10], the mask generation is slightly different. The random value r is expanded into $t = \log(\ell + 1)$ random and independent vectors $W_1, \ldots, W_t$ such that $W_i = \mathrm{E}_{K_0}(r+i)$, where K_0 is another secret key for the block cipher. Then ℓ pairwise independent and differentially uniform mask values $(S_0, S_1, \ldots, S_{\ell-1})$ are generated from the W_i, with a Gray Code or with the following method, proposed in [10]:

input: W_i, for $1 \leq i \leq t$
output: $S_0, S_1, \ldots, S_{\ell-1}$
For $i = 0$ to $\ell - 1$ do
    Let $< a_i[1], a_i[2], \ldots, a_i[t] >$ be the binary
        representation of $i + 1$
    $S_i = \bigoplus_{j=1}^{j=t} a_i[j] \cdot W_j$
end for

In [10], Jutla claims the security of IACBC in the sense of the message integrity and in the Find-Then-Guess model. However no security proof is given for this claim. In the sequel we show how to attack the scheme in the blockwise adaptive adversarial model. The attack is similar to the one described section 4: some relations between the masks values are exploited. Indeed, each mask is defined with the following relation:

$$S_{i-1} = \bigoplus_{j=1}^{j=t} a_{i-1}[j] \cdot W_j$$

for all $0 \leq i \leq l-1$ and where $< a_{i-1}[1], \ldots, a_{i-1}[t] >$ is the binary representation of i. Then in particular, we have:

$$\begin{aligned} S_1 &= W_2 \\ S_2 &= W_2 \oplus W_1 \\ S_3 &= W_3 \\ S_4 &= W_3 \oplus W_1 \end{aligned}$$

Then, for every set of mask, we have $S_1 \oplus S_2 \oplus S_3 \oplus S_4 = 0$.

During the proposed attack the attacker has access to an encryption box to mount a chosen plaintext blockwise adaptive attack. For this attack a single query to the encryption box allows to always guess correctly the message encrypted. Let us present the attacker's algorithm:

Step 1 The attacker chooses uniformly two messages of four blocks each M_0 and M_1 such that $M_0[1] = M_1[1]$, $M_0[2] = M_1[2]$, $M_0[3] = M_1[3]$ and $M_0[4] \neq M_1[4]$.

Step 2 The masks values $(S_0, \ldots, S_4)$ are generated from the random r and the vectors W_i. The encryption box encrypts r, randomly chooses a bit b and encrypts message M_b under the secret key K and transmits the ciphertext $C_b[0] \parallel C_b[1] \parallel C_b[2] \parallel C_b[3] \parallel C_b[4] \parallel C_b[5]$.

Step 3 The attacker then queries the encryption box with a message of four blocks. It first receives $C'[0]$ and outputs $M[1] = M_0[1] \oplus C'[0] \oplus C_b[0]$.

Step 4 The oracle encrypts $M[1]$ and returns $C'[1]$.

Step 5 The query continues with plaintext blocks defined by: $M[2] = M_0[2]$, $M[3] = M_0[3]$, and $M[4] = M_0[4]$.

Step 6 After having received $C'[1]$, $C'[2]$, $C'[3]$ and $C'[4]$, the adversary ends the game, receives $C'[5]$ and sends the bit $b' = 0$ if

$$C[1] \oplus C[2] \oplus C[3] \oplus C[4] = C_b[1] \oplus C_b[2] \oplus C_b[3] \oplus C_b[4] \tag{1}$$

and $b' = 1$ otherwise.

Let us look at the equality checked by the adversary. We see that if $b = 0$ we have:

$$\begin{aligned} C_b[1] \oplus C_b[2] \oplus C_b[3] \oplus C_b[4] = {} & \mathrm{E}_K(C_b[0] \oplus M_0[1]) \oplus S_1 \\ & \oplus \mathrm{E}_K(M_0[2] \oplus N_b[1]) \oplus S_2 \\ & \oplus \mathrm{E}_K(M_0[3] \oplus N_b[2]) \oplus S_3 \\ & \oplus \mathrm{E}_K(M_0[0] \oplus N_b[3]) \oplus S_4 \end{aligned}$$

where $N_b[i]$ denotes $\mathrm{E}_K(M_0[i] \oplus N_b[i-1])$ for $1 \leq i \leq 3$. Due to the choice of the test message, we also have:

$$\begin{aligned} C'[1] \oplus C'[2] \oplus C'[3] \oplus C'[4] = & \ \mathrm{E}_K(C_b[0] \oplus M_0[1]) \oplus S'_1 \\ & \oplus \mathrm{E}_K(M_0[2] \oplus N[1]) \oplus S'_2 \\ & \oplus \mathrm{E}_K(M_0[3] \oplus N[2]) \oplus S'_3 \\ & \oplus \mathrm{E}_K(M_0[4] \oplus N[3]) \oplus S'_4 \end{aligned}$$

Then if $b = 0$, since we have $S_1 \oplus S_2 \oplus S_3 \oplus S_4 = S'_1 \oplus S'_2 \oplus S'_3 \oplus S'_4 = 0$ and $N[i] = N_b[i]$ for $1 \leq i \leq 3$, equality (1) always holds.

Moreover if $b = 1$ equality (1) is never satisfied. Indeed we have $N[1] = N_b[1]$, $N[2] = N_b[2]$ and $N[3] \neq N_b[3]$ due to the special choice of the challenge messages.

Thus the attacker always guesses correctly the bit b.