Low computational cost integrity for block ciphers
José M. Sierra, Julio C. Hernández, Narayana Jayaram, Arturo Ribagorda

IT Security Laboratory, University Carlos III of Madrid, Avd. de la Universidad 30, Leganés, Madrid 28911, Spain
School of Informatics and Multimedia Technology, University of North London, UK

Available online 7 January 2004

Abstract
Encryption algorithms supply confidentiality to communications between parties. However, only under certain circumstances, encryption might also supply integrity validation. For those situations, where encryption algorithms do not supply any integrity protection, additional mechanisms must be implemented (hash functions, digital signature). In this way, many solutions have been proposed in the literature, which achieve confidentiality and integrity checking (some of them also provide authentication), however these methods usually represent a relatively high computational cost.

This paper presents a new encryption mode for block cipher algorithms, which is based on the Plaintext Cipher Block Chaining (from now on, PCBC) mode. Our new mode supplies, apart from confidentiality, fast integrity checking with a minimum computational cost. This makes it eminently suitable for ensuring data integrity in GIS systems and at the same time assuring some other GIS requirements.

© 2003 Elsevier B.V. All rights reserved.

Keywords: Data integrity in GIS; Encryption computational cost; Integrity of cipher messages; GIS confidentiality; GIS integrity

1. Introduction
A state-of-the-art application of GIS is characterized by rich data sets, and in situations where a critical functional requirement dictates that the communication protocol environment preserve data confidentiality and provide integrity with modest computational costs [7–9]. The increasing use of GIS tools has provoked a demand for more and better data (health information systems are a clear example [1]). The structure needed must be founded on information confidentiality, integrity and authentication, specially when information leaves the inter-organizational networks.

Block cipher algorithms could be considered as candidates for an encryption framework which settle the bases of confidentiality services. In what follows, after discussing salient features of block ciphers and analysing a particular mode of Cipher Block Chaining concept, we present a modification to the concept that eliminates the drawbacks of its implementation, and makes it attractive for use in GIS systems because includes integrity checking with a minimum computational cost.

A block cipher algorithm is a type of encryption algorithm where the cleartext is divided into blocks of the same length, which are ciphered independently with the same ciphering key and algorithm. Examples of block ciphers are: the widely known as Data Encryption Standard (DES) [2], which has been the standard encryption algorithm for the United States during more than 30 years, and also its successor Advance Encryption Standard (AES) [3].

Block ciphers usually implement ciphering modes, which combine the blocks with certain basic operations and feedbacks. Operations used must be
computationally fast and efficient and must increase the security of the ciphertext, making its cryptanalysis more difficult. In any case, confidentiality of the ciphertext is only based on the robustness of the cipher algorithm but never in the ciphering mode.

2. Cipher modes

The simplest cipher mode is the Electronic Code Book (widely known as ECB) that only consists on ciphering each cleartext block independently (there is no feedback among blocks and no additional operations are done). Although this mode is very simple, its simplicity allows the parallelization of the ciphering process, which is very useful in terms of computational cost associated to the encryption. Obviously the ECB does not supply any integrity validation, and hence modifications, intentional or not, of the ciphertext.

Arguably the most widely used cipher mode is the Cipher Block Chaining (known as CBC). In this case, each cleartext block is operated, by the \texttt{XOR} function, with the ciphertext produced in the previous block. The result of this operation is ciphered and, as the result of this, it is obtained the ciphertext block (the process is shown in Fig. 1).

In the CBC mode, by only adding one additional \texttt{XOR} operation to each block, two identical cleartext blocks produce different ciphertext blocks, and this fact improves security of the ciphertext, because it makes its cryptanalysis harder. To decrypt a ciphered message the process must be inverted (see Fig. 2).

Although the CBC mode is used as default in many communications protocols, it cannot supply any mechanism for the integrity verification. In this mode, one bit alteration in certain ciphertext block will produce, when the message is decrypted, errors in two cleartext blocks.

Our problem is not only to find out a mechanism for the verification of the integrity of messages, but also to find a very efficient mechanism that does not delay the encryption process. With this in view the Massachusetts Institute of Technology designed a new cipher mode that solved this problem. This mode was called Plaintext Cipher Block Chaining (from now on PCBC) which was based on the CBC, but adding a new \texttt{XOR} operation to each block. This modification makes possible to detect integrity alterations in ciphertexts. We will cover the PCBC mode in our next section.

3. Plaintext Cipher Block Chaining

The PCBC mode was created for the Kerberos Version 4 \cite{6}. It was based on the CBC mode, but it added a new \texttt{XOR} operation. In fact what PCBC does is to combine the previous plaintext block with the current ciphertext block and also with the previous ciphertext block obtained. Fig. 3 shows the process for encryption and decryption using the PCBC mode.

Analysing the algorithm, what PCBC really does is, to propagate any alteration of the ciphertext to the last block. In this way, by adding to the end of the message a fixed last cleartext block, the destination of the communication can verify the integrity of the message,
simply by decrypting the last block and checking that the fixed cleartext has not changed.

Regrettably, the PCBC mode has not been formally published as standard because there is a vulnerability discovered by Kohl in 1990 [5]. This vulnerability permits that if two ciphertext blocks are swapped, then the result of the decryption of the last block still yields the correct fixed plaintext. Though the practical consequences of this flaw are not obvious, PCBC was replaced by CBC-MD5 mode in Kerberos Version 5.

Next algebraic expressions describe the ciphertext in the PCBC mode. Let us consider $P_i$ the $i$th cleartext block, $C_i$ the $i$th ciphertext block, $E_k$ (and $D_k$ for decryption) the symmetric cipher algorithm, which use the key $K$:

$$C_{i-1} = E_k(P_{i-1} \oplus P_{i-2} \oplus C_{i-2}),$$
$$C_i = E_k(P_i \oplus P_{i-1} \oplus C_{i-1}),$$
$$C_{i+1} = E_k(P_{i+1} \oplus P_i \oplus C_i).$$

Equations to obtain the cleartext are:

$$P_{i-1} = D_k(C_{i-1}) \oplus P_{i-2} \oplus C_{i-2},$$
$$P_i = D_k(C_i) \oplus P_{i-1} \oplus C_{i-1},$$
$$P_{i+1} = D_k(C_{i+1}) \oplus P_i \oplus C_i.$$  \hspace{1cm} (1)

Operating with (1) it is possible to obtain:

$$P_{i+1} = D_k(C_{i+1}) \oplus D_k(C_i) \oplus P_{i-1} \oplus C_{i-1} \oplus C_i,$$
$$P_{i+1} = D_k(C_{i+1}) \oplus D_k(C_i) \oplus D_k(C_{i-1}) \oplus P_{i-2} \oplus C_{i-2} \oplus C_{i-1} \oplus C_i.$$ \hspace{1cm} (2)

If we swap the ciphertext block $C_{i-1}$ and $C_i$, now the cleartext is:

$$P_{i-1} = D_k(C_i) \oplus P_{i-2} \oplus C_{i-2},$$
$$P_i = D_k(C_{i-1}) \oplus P_{i-1} \oplus C_i,$$
$$P_{i+1} = D_k(C_{i+1}) \oplus P_i \oplus C_{i-1}.$$ \hspace{1cm} (3)
Operating with these expressions, it is possible to obtain:

\[ P'_{i+1} = D_k(C_{i+1}) \oplus D_k(C_{i-1}) \oplus P'_{i} \oplus C_{i} \oplus C_{i-1}, \]

\[ P'_{i+1} = D_k(C_{i+1}) \oplus D_k(C_{i-1}) \oplus D_k(C_{i}) \oplus P_{i-2} \oplus C_{i-2} \oplus C_{i-1}. \] (4)

It is clear that expressions (2) and (4) are identical, due to the features of the XOR operation. This explains how, when swapping two ciphered blocks, the following ones will not suffer any modification. This weakness of the PCBC, which only appears under very specific situations, does not destroy the foundations of this mode. In fact, the inclusion of one additional XOR operation gives message integrity verification at a very similar computational cost, which is definitely a good deal.

Our next section describes how the commented PCBC flaw could be solved maintaining the computational cost restrictions assumed by the PCBC mode authors. We have developed a new algorithm, called memory-PCBC, which follows the commented restrictions.

### 4. New memory-PCBC algorithm

In this work we present a modification of the PCBC algorithm, which solved the commented flaw and supplies an efficient integrity validation. Our new algorithm is based on adding a memory that stores information of the past ciphertext blocks. With this modification it is possible to propagate any error to the last block of the message preserving the
computational cost restrictions of the PCBC. Our algorithm includes a new variable that stores information from the past ciphertext blocks. Computationally, our algorithm means a very slight increment because, for each block, we only add one XOR operation, and hence, we maintain the performance features of the original PCBC but detecting any integrity violation.

In our algorithm, we use a memory \( M \) whose length \( n \) is equal to the length of the cipher blocks. We have divided \( M \) into two variables of length \( n/2 \), these variables represent the left and right part of the memory. Fig. 4 shows the encryption and decryption process of the M-PCBC.

The \( M_i \) will be initialized with the IV, also used in the original PCBC, and every block \( M_i \) will be formed by information from the previous ciphertext and from the previous memory content. Both contents will be mixed according to the following expressions:

\[
M_{i+1}^L \leftarrow M_{i-1}^L \oplus M_{i-1}^R, M_{i+1}^R \leftarrow C_{i-1}^L \oplus C_{i-1}^R.
\]

Fig. 5 shows this process graphically.

Although there are two XOR operations, those are of \( n/2 \) length, which computationally represents an XOR operation of length \( n \).

The M-PCBC, in contrast with the original PCBC and also with the standard CBC, does not use the ciphertext to be XOR red with other blocks. In our algorithm, we only use half of the memory to store information of the previous ciphertext. We have chosen to digest the previous ciphertext block into the memory, instead of using the previous ciphertext, due to the avalanche effect (all its bits depend on the ciphering key and also on the cleartext) and also because its entropy is considerably higher.

Furthermore, considering the commented flaw of the original PCBC, where if we swap two contiguous blocks it is possible to avoid the integrity checking, our M-PCBC will detect this type of violations because algorithm will avoid the duplication of any past memory content. To prove this fact, let us analyse the new algorithm M-PCBC and compare it with the previous analysis of the PCBC algorithm in \([2,4]\).

Next algebraic expressions define M-PCBC ciphering mode:

\[
C_{i+1}^L = E_k(P_i \oplus R_{i-1} \oplus P_{i-1}),
\]

\[
C_i = E_k(P_i \oplus R_i \oplus P_i),
\]

\[
C_{i+1} = E_k(P_i \oplus R_{i+1} \oplus P_{i+1}).
\]

Considering the \( M_t \) calculation as a \( G \) function, which depends on the previous memory content and the previous ciphertext, the equations to obtain the cleartext
from the ciphertext are:

\[ P_{-1} = P_{-2} \oplus D(C_{-1}) \oplus G(R_{-2}, C_{-2}), \]
\[ P_{i} = P_{i-1} \oplus D(C_{i}) \oplus G(R_{i-1}, C_{i-1}), \]
\[ P_{i+1} = P_{i} \oplus D(C_{i+1}) \oplus G(R_{i}, C_{i}). \]  

(5)

Operating with the expression [5] and resolving the \( P_{i} \) and \( P_{i+1} \), we obtain:

\[ P_{i+1} = P_{i-1} \oplus D(C_{i}) \oplus G(R_{i-1}, C_{i-1}) \]
\[ \quad \oplus D(C_{i+1}) \oplus G(R_{i}, C_{i}), \]
\[ P_{i+1} = P_{i-2} \oplus D(C_{i-1}) \oplus D(C_{i}) \oplus D(C_{i+1}) \]
\[ \quad \oplus G(R_{i-2}, C_{i-2}) \oplus G(R_{i-1}, C_{i-1}) \]
\[ \quad \oplus G(R_{i}, C_{i}). \]

Resolving the \( G \) function to obtain the memory contents we have:

\[ P_{i+1} = P_{i-2} \oplus D(C_{i-1}) \oplus D(C_{i}) \oplus D(C_{i+1}) \]
\[ \quad \oplus D(C_{i+1}) \oplus G(R_{i}, C_{i}) \]
\[ \quad \oplus D(C_{i}) \oplus G(R_{i-1}, C_{i-1}) \]
\[ \quad \oplus G(R_{i-2}, C_{i-2}) \oplus G(R_{i-1}, C_{i-1}) \]
\[ \quad \oplus G(R_{i}, C_{i}). \]  

(6)

If we compare expressions (2) and (6), which show the dependence of the PCBC and the M-PCBC, both depend on \( P_{i-2}, C_{i-2}, C_{i-1}, C_{i} \) and \( D_{2}(C_{i+1}), D_{3}(C_{i}), D_{4}(C_{i}) \). However \( P_{i+1} \) in (6) also depends on \( M_{i-2}, \beta_{M_{i-1}}. \)

Furthermore, let suppose that, in M-PCBC, we swap \( C_{i-1} \) and \( C_{i} \), now the expressions are:

\[ P'_{i-1} = P_{i-2} \oplus D(C_{i}) \oplus G(R_{i-2}, C_{i-2}), \]
\[ P'_{i} = P'_{i-1} \oplus D(C_{i-1}) \oplus G(R_{i-1}, C_{i}), \]
\[ P'_{i+1} = P'_{i} \oplus D(C_{i+1}) \oplus G(R_{i}, C_{i-1}). \]  

(7)

Developing these expressions and substituting \( P'_{i+1} \) in (7), we have:

\[ P'_{i+1} = P_{i-2} \oplus D(C_{i}) \oplus D(C_{i-1}) \oplus D(C_{i+1}) \]
\[ \quad \oplus G(R_{i-2}, C_{i-2}) \oplus G(R_{i-1}, C_{i-1}), \]
\[ P'_{i+1} = P_{i-2} \oplus D(C_{i}) \oplus D(C_{i-1}) \oplus D(C_{i+1}) \]
\[ \quad \oplus G(R_{i-2}, C_{i-2}) \oplus G(R_{i-1}, C_{i}) \]
\[ \quad \oplus G(R_{i}, C_{i-1}). \]

Resolving the \( G \) function to obtain the memory contents we have:

\[ P'_{i+1} = P_{i-2} \oplus D(C_{i}) \oplus D(C_{i-1}) \oplus D(C_{i+1}) \]
\[ \quad \oplus D(C_{i+1}) \oplus G(R_{i}, C_{i}) \]
\[ \quad \oplus D(C_{i}) \oplus G(R_{i-1}, C_{i-1}) \]
\[ \quad \oplus G(R_{i-2}, C_{i-2}) \oplus G(R_{i-1}, C_{i}) \]
\[ \quad \oplus G(R_{i}, C_{i-1}). \]  

(8)

Expressions (6) and (8) are not equal because both parts of the memory contents are different. This is due to function \( G \), which is not commutative, and because the memory maintain digested information about the last ciphertext and past memory content.

5. Conclusions

Different organizations are occupied in increased business activities over the networks. Organizations from sectors such as automotive, healthcare, government, banking, education and others are particularly interested on having information structured in GIS systems. GIS have the potential to be more menacing to security than other information technologies due to the possible combination of powerful data integration and analysis capabilities with data that local in nature [7].

Information security is a complex area where, even given powerful mechanisms and tools, there is no certainty that these can be applied or used in all situations. This is owing to the impact of these techniques on certain limited resources of the information systems (computational, network, storage, etc.). For this reason while designing security services the constraints on mechanisms and tools must be taken into account and one should be aware of these features and carefully select those mechanisms which not provide adequate security, but produce minimal impact on resources as referred to above.

In this paper, we have proposed a new encryption mode for symmetric cipher algorithms. This type of ciphers is widely used and, for that reason, implications of our work will be appreciated in environments such as the GIS. In fact, commonly it is needed to use two different mechanisms to accomplish confidentiality and integrity; however, with our algorithm M-PCBC it is possible to achieve both only using encryption. Our algorithm can be helpful not only when the CPU usage is important but also when there are restrictions on the amount of information that can be transmitted.
There are several ciphering modes for block ciphers. Basically what they try is to improve the security level of the ciphertext by adding some simple low computational cost operations. Consequently, design of ciphering modes is quite a complex task because it is limited to use of very simple operations, such as the XOR. Those foundations were taken into account in the design of original PCBC for Kerberos and for this reason we have chose PCBC to build our algorithm. There are other algorithms that supply message integrity as a ciphering mode (one of the most interesting is [4]), however its complexity and computational cost are far from original PCBC and hence that of our M-PCBC.

Computational cost of M-PCBC is very similar to initial PCBC proposal, because we only include one more XOR operation to the algorithm. But low computational cost is not the only advantage of our algorithm. M-PCBC solves commented PCBC flaw and supplies a stronger ciphering mode. Our proposal is more sensible to integrity violations; in fact, one bit alteration in certain ciphertext block will be amplified to the rest of the plaintext blocks and will be also stored into the memory, both things make easier its detection. The low computational characteristics coupled with the strong ciphering mode makes this algorithm attractive for data integrity situations in GIS.

Improvements of M-PCBC can be associated with the inclusion of certain modifications to make it a suitable candidate for parallelization. Other proposals of parallelizable ciphering modes involve the important computational cost augment, which also implies that a future parallelizable M-PCBC algorithm will also involve a computational cost raise.

References


José M. Sierra is an Associated Professor at the Computer Science Department of the University Carlos III of Madrid. His research work is centred in the area of the Internet Security and, in this area, he is working and researching. He has participated in numerous projects of investigation, and has published articles in journals related with the Security in the Technologies of the Information.

Julio C. Hernandez is a Senior Lecturer at the Computer Science Department of the University Carlos III of Madrid. He is Bachelor in Mathematics at the Universidad Complutense de Madrid, later he worked as system and network administrator and consultant of security and he obtained the University degree of Specialist in Information Coding and Network Security by the University of Valladolid. Currently he is integrated into the IT Security Group of the University Carlos III of Madrid.

Arturo Ribagorda is an Telecommunications Engineer and PhD in Computer Science at the Politecnical University of Madrid. At the moment he is Full Professor and Head of the of the Computer Science Department of the University Carlos III of Madrid. His research area is the Information Technologies Security, field in which he worked in numerous research projects, national and international. He has published more than 40 papers in national magazines and international journals. He has been invited chair in numerous conferences and has written three books. In addition, he has been, and currently he is, evaluator in European Research Programs (Advanced Transport Telematics and ESPRIT).