ROYAL HOLLOWAY, UNIVERSITY OF LONDON
Information Security Group

Cryptanalysis of the EPBC authenticated encryption mode

Chris Mitchell

Royal Holloway, University of London

http://www.isg.rhul.ac.uk/~cjm

ROYAL HOLLOWAY, UNIVERSITY OF LONDON
Information Security Group

Agenda

- **Introduction**
- Simultaneous confidentiality and integrity
- Attacking EPBC
- Completing the attack

Simultaneous encryption and integrity

- Both confidentiality and integrity are often required.
- Indeed, encrypting without integrity protection is now known to be dangerous (variety of attacks).
- One simple way to provide both services is the *encrypt-then-MAC* model where we encrypt the message and then compute a MAC, using two distinct keys.
- This is very effective (if used with care), but each block of data is processed twice.

ROYAL HOLLOWAY, UNIVERSITY OF LONDON
Information Security Group

Add-redundancy-and-encrypt model

- To avoid the extra work of double processing, one widely discussed alternative to *encrypt-then-MAC* is the *add-redundancy-and-encrypt* model.
- Here, predictable redundancy is added to the plaintext (e.g. a fixed block at the end) prior to encryption, and the receiver checks for the presence of the redundancy after decryption.

Shortcomings of model

- The encryption method needs to be chosen carefully (e.g., a stream cipher is bad news)!
- So does the method of adding redundancy.
  - Suppose the ‘fixed block at the end’ method is used.
  - Obvious dangers arise if the fixed block arises by chance in the middle of the plaintext!
- Despite these dangers, the technique has often been advocated.

EPBC mode

- One major problem with the add-redundancy-then-encrypt approach is that commonly used encryption modes are not appropriate.
- That is, if a mode like CBC is used, then relatively simple forgery attacks are possible (as we show).
- We consider a mode specially designed for use with add-redundancy-then-encrypt, namely EPBC, and show that this mode too is subject to forgery attacks.

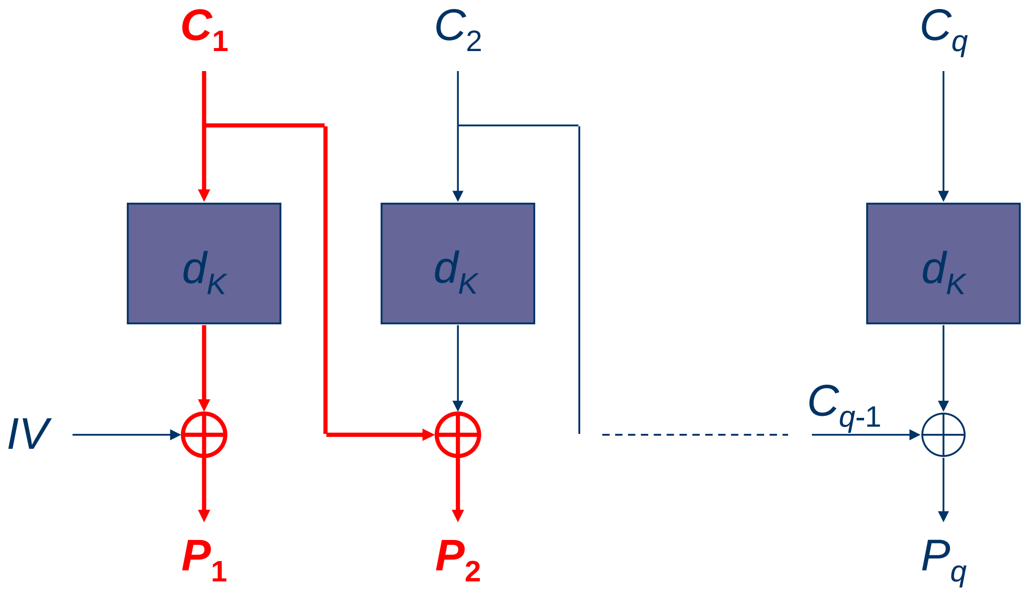
Agenda

- Introduction
- **Simultaneous confidentiality and integrity**
- Attacking EPBC
- Completing the attack

CBC mode = no good!

- Having decided to use *add-redundancy-and-encrypt*, the encryption method needs to be chosen.
- It is not hard to see that CBC mode is completely inappropriate.
- This is because ciphertext errors only propagate in a very limited way.
- That is, changing ciphertext block C_i only affects P_i and P_{i+1}.

CBC decryption – error propagation



EPBC mode

- EPBC (Efficient error-Propagating Block Chaining) was proposed by Zúquete and Guedes in 1997.
- It is a mode of operation in which ciphertext errors **propagate in an unlimited way**.
- Designed as an improvement of a mode called IOBC (Recacha, 1996).
- Uses an n-bit block cipher where n is even (assume n=2m).

EPBC mode operation

- Uses two n-bit secret IVs: F_0, G_0.
- To encrypt plaintext $P_1, P_2, \ldots, P_t$:
  - perform the following for $i = 1, 2, \ldots, t$:
    - $G_i = P_i \oplus F_{i-1}$
    - $F_i = e_K(G_i)$
    - $C_i = F_i \oplus g(G_{i-1})$ [except for i=1: $C_1 = F_1 \oplus G_0$]

where $\oplus$ denotes bit-wise exclusive or, and g is a function mapping an n-bit block to an n-bit block.

The function *g*

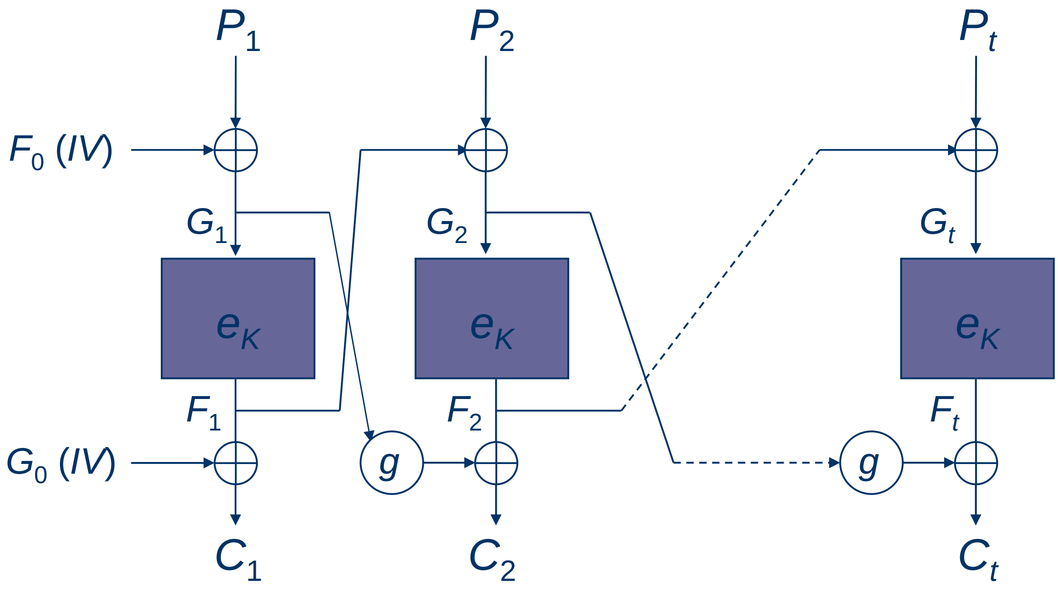
- Suppose X is an n-bit block, where $X = L||R$, and L and R are m-bit blocks.
- Then:

$$g(X) = (L \vee \sim R) \,||\, (L \wedge \sim R)$$

where $\vee$ denotes bit-wise inclusive or, $\wedge$ denotes bit-wise logical and, and ~ denotes logical negation (changing every zero to one and vice versa).

- Note that g is not one-to-one. [This is the only change between OPBC to EPBC: IOBC uses a one-to-one function g].

EPBC encryption (also IOBC)



An observation

- To launch a forgery attack, it would appear to be necessary to have knowledge of the ‘internal’ values of F_i and G_i.
- However, since these values are never transmitted (and F_0 and G_0 are assumed to be secret), attacking this mode would appear to be difficult.
- Moreover, *g* is deliberately chosen to be not one-to-one to thwart known-plaintext based forgery attacks which apply to long messages encrypted using IOBC.

Agenda

- Introduction
- Simultaneous confidentiality and integrity
- **Attacking EPBC**
- Completing the attack

Objective of attack

- We assume that the *add-redundancy-and-encrypt* model is being used with EPBC.
- We also assume that the method of adding redundancy is to add a fixed block to the end of the message.
- The objective is to take a valid ciphertext and use this to construct another 'forged' ciphertext which will have the correct redundancy when decrypted.

Observation regarding *g*

- Suppose $g(X) = L'||R'$, where $L' = (\lambda'_1, \lambda'_2, .., \lambda'_m)$ and $R' = (r'_1, r'_2, .., r'_m)$.
- Then, for every i, if $\lambda'_i = 0$, then $r'_i = 0$.
- To see this, suppose $X = L||R$, where $L = (\lambda_1, \lambda_2, .., \lambda_m)$ and $R' = (r_1, r_2, .., r_m)$.
- If $\lambda'_i = 0$ for some i, then, since $\lambda'_i = \lambda_i \vee \sim r_i$, we know immediately that $\lambda_i = 0$ and $r_i = 1$. Hence $r'_i = \lambda_i \wedge \sim r_i = 0$.
- That is, pairs (λ'_i, r'_i) can never equal (0, 1).

A more general observation

- Using the same notation, if (λ_i, r_i) is in the set A, then (λ'_i, r'_i) must be a member of the set B, where the possibilities for the sets A and B are now given.
- Unless $|A| = 1$, given a random set A of a certain size, the expected size of B is always smaller than $|A|$.

The sets *A* and *B*

A (set of input pairs)	***B*** (set of output pairs)
{00, 01, 10, 11}	{00, 10, 11}
{01, 10, 11} {00, 10, 11} {00, 01, 11} {00, 01, 10}	{00, 10, 11} {10, 11} {00, 10} {00, 10, 11}
{10, 11} {01, 11} {01, 10} {00, 11} {00, 10} {00, 01}	{10, 11} {01, 11} {00, 11} {10} {10, 11} {00, 10}
{11} {10} {01} {00}	{10} {11} {00} {10}

Using the observation I

- Our objective is to use knowledge of known plaintext/ciphertext pairs (P_i, C_i) to learn pairs (F_i, G_i).
- Suppose we know *s* consecutive pairs, i.e. we know:

$$(P_j, C_j), (P_{j+1}, C_{j+1}), \ldots, (P_{j+s-1}, C_{j+s-1}).$$

where we suppose $j > 1$.

Using the observation II

- We know:

$$C_j = F_j \oplus g(G_{j-1})$$

- We also know that if $g(G_{j-1}) = L'||R'$, where $L' = (\lambda'_1,\lambda'_2,..,\lambda'_m)$ and $R' = (r'_1,r'_2,..,r'_m)$, then (λ'_i, r'_i) can never equal (0, 1) for any i.
- Hence, knowledge of C_j gives some knowledge about F_j.
- Specifically we know that certain bit pairs cannot occur in F_j, where each bit pair contains a bit from the left half and the corresponding bit from the right half.

Using the observation III

- We also know:

$$G_{j+1} = P_{j+1} \oplus F_j$$

- Hence knowledge of forbidden bit pairs in F_j, combined with knowledge of P_{j+1}, gives us knowledge of forbidden bit pairs in G_{j+1}.
- This means we know of even more (potentially) forbidden bit pairs in $g(G_{j+1})$.

Using the observation IV

- Since we know:
$$C_{j+2} = F_{j+2} \oplus g(G_{j+1})$$
and we know C_{j+2}, this gives us even more forbidden bit pairs in F_{j+2}, and so on.
- For sufficiently large w, we hope that we know F_{j+2w} for certain.
- This immediately gives complete knowledge of G_{j+2w+1}, using knowledge of P_{j+2w+1}.
- I.e. we have complete knowledge of all F_{j+2w} and G_{j+2w+1} for all sufficiently large w.

A side remark

- In our discussion we have not used all the available knowledge.
- In fact we only use knowledge of C_j, C_{j+2}, C_{j+4}, … and P_{j+1}, P_{j+3}, P_{j+5}, …
- We also only learn information about F_j, F_{j+2}, F_{j+4}, … and G_{j+1}, G_{j+3}, G_{j+5}, …
- However, we now repeat the process starting with F_{j+1}, using all the rest of the information we have.

How big is sufficiently large?

- Consider any pair of bit positions: (i, $i+m$).
- Returning to our previous argument, we know that $g(G_{j-1})$ cannot have (0, 1) in these two bit positions.
- Hence, we know that the pair of bit positions in $F_j = C_j \oplus g(G_{j-1})$ can only take three of the possible four values.
- Precisely which three possibilities will depend on C_j, which should look random.

How big II?

- Hence we know that the two bit positions in G_{j+1} can only take three of the possible four values.
- The possibilities for the two bit positions in $g(G_{j+1})$ will depend on which three pairs are possible (using our table for the sets A and B).
- That is, there is a 50% chance that we will know that the two bit positions in $g(G_{j+1})$ have only two possible values, and a 50% chance that there are 3 possible values.

How big III?

- Using standard probabilistic arguments for stochastic processes, the probability that there will only be a single possibility for the bit pair after *v* iterations of the above process is equal to the top right entry in the *v*th power of the following 4 by 4 matrix:

$$\begin{pmatrix} 0 & 1 & 0 & 0 \\ 0 & 1/2 & 1/2 & 0 \\ 0 & 0 & 5/6 & 1/6 \\ 0 & 0 & 0 & 1 \end{pmatrix}$$

How big IV?

- For *v* = 10, this is 0.710.
- For *v* = 20, this is 0.953.
- That is, after 20 iterations, i.e., if we know 40 consecutive plaintext/ciphertext pairs, we will know for certain around 95% of the bit pairs.
- I.e., if *m*=64, we will know for certain around 120 of the 128 bits.
- There will only be a small number of possibilities for the other bit pairs.

Agenda

- Introduction
- Simultaneous confidentiality and integrity
- Attacking EPBC
- **Completing the attack**

What else needs to be done?

- Once we know some values of F_i and G_i, we need to use these values to construct a forgery.
- This is straightforward, as we now show.
- We suppose that the added redundancy prior to encryption is a fixed *n*-bit block, i.e. the final *n*-bit block of a plaintext message is equal to a fixed block, *V*.
- The presence (or absence) of this block is used by a decrypter to check that a message is valid (or not).

Resources for attack

- We suppose that an attacker has the first *s* blocks of an encrypted message $C_1, C_2, \ldots, C_s$, for which he/she knows the internal value G_s.
- We suppose the attacker also knows the final two blocks (C'_{u-1}, C'_u) of an encrypted message for which the attacker knows the internal value G'_{u-2}. [NB: if P'_u is the final plaintext block of this message, then $P'_u = V$.]
- We suppose these two part ciphertexts have been encrypted using the same key *K*. [These two part ciphertexts could be the first *s* blocks and the final 2 blocks of a longer encrypted message].

A forged message

- We now define a 'forged' ciphertext message:

$$C^*_1, C^*_2, \ldots, C^*_{s+2}$$

- where

$$C^*_i = C_i \quad (1 \leq i \leq s);$$
$$C^*_{s+1} = C'_{u-1} \oplus g(G'_{u-2}) \oplus g(G_s);$$
$$C^*_{s+2} = C'_u.$$

- When this forged message is decrypted, the final block will be $P'_u = V$.

Encrypt-then-MAC model

- There seem to be too many problems with the *add-redundancy-and-encrypt* model to be able to recommend it.
- *Encrypt-then-MAC* seems much safer, and is provably secure.
- However even this approach needs to be implemented with care; in particular, a decrypter must not attempt to decrypt a message if the MAC check fails.

Combined encryption/integrity modes

- There are alternatives to *encrypt-then-MAC*.
- Of particular interest is the Offset CodeBook (OCB) mode, due to Rogaway, Bellare, Black and Krovetz (2001), and a revised OCB v2.0 more recently released.
- These block-cipher-based modes only require each plaintext block to be processed once, and have a complexity-theoretic 'proof of security' (based on the assumption that the block cipher is a pseudo-random permutation family).

Standards

- OCB v2.0, together with other carefully specified ways of combining encryption and MACing, are in the process of being standardised.
- One such standard will be ISO/IEC 19772 (currently at Committee Draft stage).

Further reading

- The attack described in this talk is given in:
  - C. J. Mitchell, *Cryptanalysis of the EPBC authenticated encryption mode*. [Copies available from the author].
- For more information on cryptography standards (including MACs, modes, etc.) see:
  - A. Dent and C. Mitchell, *User's guide to cryptography and standards*. Artech House, 2005.
- The standard reference for much of crypto (even though it is now 10 years old) remains:
  - A. Menezes, P. van Oorschot and S. Vanstone, *Handbook of Applied Cryptography*, CRC Press, 1997.