Parallelizable Encryption Mode with Almost Free Message Integrity

Charanjit S. Jutla
IBM T.J. Watson Research Center,
Yorktown Heights, NY 10598

In this document we propose a new mode of operation for symmetric key block cipher algorithms. The main feature distinguishing the proposed mode from existing modes is that along with providing confidentiality of the message, it also provides message integrity. In other words, the new mode is not just a mode of operation for encryption, but a mode of operation for authenticated encryption. As the title of the document suggests, the new mode achieves the additional property with little extra overhead, as will be explained below.

The new mode is also highly parallelizable. In fact, it has critical path of only two block cipher invocations. By one estimate, a hardware implementation of this mode on a single board (housing 1000 block cipher units) achieves terabits/sec (10^{12} bits/sec) of authenticated encryption. Moreover, there is no penalty for doing a serial implementation of this mode.

The new mode also comes with proofs of security, assuming that the underlying block ciphers are secure. For confidentiality, the mode achieves the same provable security bound as CBC. For authentication, the mode achieves the same provable security bound as CBC-MAC.

The new parallelizable mode removes chaining from the well known CBC mode, and instead does an input whitening (as well an output whitening) with a pairwise independent sequence. Thus, it becomes similar to the ECB mode. However, with the input whitening with the pairwise independent sequence the new mode has provable security similar to CBC (Note: ECB does not have security guarantees like CBC). Also, the output whitening with the pairwise independent sequence guarantees message integrity.

The pairwise independent sequence can be generated with little overhead. In fact, the input and output whitening sequence need only be pairwise differentially uniform, which is a weaker property than pairwise independence, as explained in the details below. The weaker pairwise differentially uniform sequence can be generated

with even lesser overhead.

The parallelizable mode comes in two flavors. These flavors refer to how the pairwise differentially uniform sequence is generated. In one mode, we just use a pairwise independent sequence generated by a subset construction. In another mode, the pairwise differentially uniform sequence is generated by $(a * i)$ modulo a fixed prime number. There will be one standard prime number for each bit-size block cipher. Thus, for 64 bit block ciphers the prime could be $2^{64} - 257$. For 128 bit block ciphers, the prime could be $2^{128} - 159$.

The modes are described below in more detail. For proofs of security see proceedings of Eurocrypt 2001 (also http://eprint.iacr.org/2000/039.ps).

We first give definitions of pairwise independence and related concepts. Then we describe the parallelizable mode using the *algebraic* construction $a*i$ modulo a fixed prime. Finally, we describe the mode using only exclusive-or operations.

1. Definitions

Definition 1 (pair-wise independence) A sequence of uniformly distributed n-bit random numbers $s_1, s_2, ..., s_m$, is called *pair-wise independent* if for every pair $i, j, i \neq j$, and every pair of n bit constants c_1 and c_2, probability that $s_i = c_1$ and $s_j = c_2$ is 2^{-2n}.

Definition 2 (pair-wise differentially-uniform) A sequence of uniformly distributed n-bit random numbers $s_1, s_2, ..., s_m$, is called *pair-wise differentially-uniform* if for every pair $i, j, i \neq j$, and every n bit constant c, probability that $s_i \oplus s_j$ is c is 2^{-n}.

It is a fact that a pair-wise independent uniformly distributed sequence is also pair-wise differentially uniform.

Definition 3 (pair-wise differentially-uniform in GFp) A sequence of random numbers $s_1, s_2, ..., s_m$ uniformly distributed in GFp, is called *pair-wise differentially-uniform in* GFp if for every pair $i, j, i \neq j$, and every constant c in GFp, probability that $(s_i - s_j)$ mod p is c is $1/p$.

A sequence of m pair-wise independent numbers can be generated from about $\log m$ independent random numbers by a subset construction. The subset construction only involves exclusive-or operations.

A pair-wise independent sequence can also be generated by an algebraic construction in GFp, by using two independent random numbers a and b in GFp. The sequence is given by $s_i = (a + i * b)$ mod p.

A pair-wise differentially uniform in GFp sequence can be generated from only a single random number a in GFp by defining $s_i = (i * a)$ mod p.

These definitions have been given here only to explain the general principles. In the following description of the new mode, the sequence used is a subtle variation of definition 3. For proofs of security, see the aforementioned references.

2. Integrity Aware Parallelizable Mode (IAPM) using a prime number

Let n be the block size of the underlying block cipher. We will restrict our attention to $n = 128$ in this paper. If the block cipher requires keys of length k, then this mode requires two independent keys of length k. Let these keys be called $K0$ and $K1$. From now on, we will use f_K to denote the encryption function under key K.

The message to be encrypted P, is divided into blocks of length n each. Let these blocks be $P_1, P_2, ..., P_{m-1}$. As in CBC, a random initial vector r of length n bits is chosen. The vector r need not be chosen randomly, as long as it is unique for each message. This random vector is used to generate a new random vector a using the block cipher and key K0, which in turn is used to prepare $m + 1$ new pairwise differentially uniform vectors $S_0, S_1, ..., S_m$.

Let $p = 2^{128} - 159$. The number p is known to be a prime. This prime will be fixed for all invocations of this mode using block ciphers of block size 128 bit. For 64-bit ciphers $p = 2^{64} - 257$ is recommended.

Now, the sequence $S_0, S_1, ...S_m$ is generated by the following procedure:

procedure pairwise_diff_uniform_sequence(*in* $r, m, K0$; *out* S)
{
$a = f_{K0}(r)$
if $(a \geq (2^{128} - 159))$ $a = (a + 159) \bmod 2^{128}$
$S_0 = a$
for $i = 1$ to m do
$S_i = (S_{i-1} + a) \bmod 2^{128}$
if $(a > S_i)$ $S_i = S_i + 159$
end for
}

The condition $(a > S_i)$ is equivalent to 128-bit integer addition overflow in the previous step. Note that we do not reduce modulo p if $(S_{i-1} + a) < 2^{128}$, but we do compensate by 159 if $(S_{i-1} + a) \geq 2^{128}$, as in the latter case, $(S_{i-1} + a)$ mod p $= S_{i-1} + a - (2^{128} - 159) = (S_{i-1} + a - 2^{128}) + 159$.

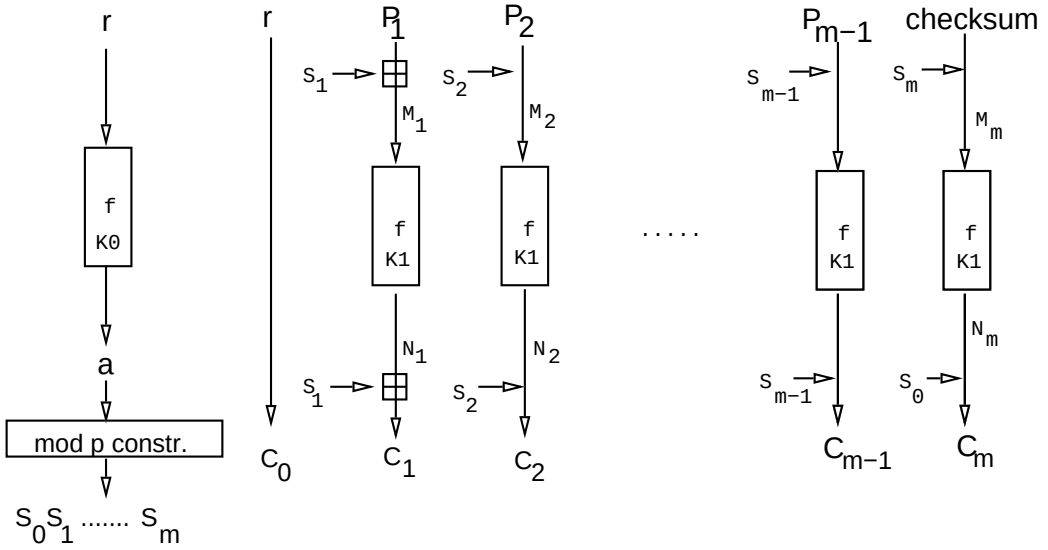

Figure 1: Integrity Aware Parallelizable Mode (IAPM)

In this mode, the input and output whitening is done by 128-bit integer addition. The ciphertext message $C =< C_0, C_1, ..., C_m >$ is generated as follows (see fig 1):

$C_0 = r$
for $i = 1$ to $m - 1$ do
$M_i = (P_i + S_i) \bmod 2^{128}$
$N_i = f_{K1}(M_i)$
$C_i = (N_i + S_i) \bmod 2^{128}$
end for
checksum $= P_1 \oplus P_2 \oplus ... \oplus P_{m-1}$
$M_m = (\text{checksum} + S_m) \bmod 2^{128}$
$N_m = f_{K1}(M_m)$
$C_m = (N_m + S_0) \bmod 2^{128}$

Note that for computing the checksum we use xor instead of addition modulo 2^{128}. The scheme is secure even if the checksum is computed by a modulo 2^{128} sum, but for the standard we prefer that the checksum be computed by an xor-sum. Note that S_0 is used in the last step.

The above scheme is invertible. The inversion process yields blocks $P_1, P_2, ..., P_m$. The decrypted plaintext is $< P_1, P_2, ..., P_{m-1} >$. Message integrity is verified by

checking

$$P_m = P_1 \oplus P_2 \oplus ... \oplus P_{m-1}$$

Here is the pseudo-code for decryption:

$r = C_0$
invoke pairwise_diff_uniform_sequence$(r, m, K0, S)$;
for $i = 1$ to $m - 1$ do
    $N_i = (C_i - S_i) \bmod 2^{128}$
    $M_i = f_{K1}^{-1}(N_i)$
    $P_i = (M_i - S_i) \bmod 2^{128}$
end for
checksum $= P_1 \oplus P_2 \oplus ... \oplus P_{m-1}$
$N_m = (C_m - S_0) \bmod 2^{128}$
$M_m = f_{K1}^{-1}(N_m)$
$P_m = (M_m - S_m) \bmod 2^{128}$
Integrity $\equiv$ $(P_m ==$ checksum$)$

3. IAPM with only xor operations

The mode described above uses integer addition. We now describe a similar mode in which the only operations are block cipher invocations and exclusive-or operations. In particular, the pairwise differentially uniform sequence is generated using a subset construction. Actually, this sequence has the stronger property of pairwise independence. The subset construction is also optimized using Gray code (http://hissa.nist.gov/dads/HTML/graycode.html). The penalty one has to pay in this mode is that instead of generating one extra vector a as described in the previous section, one now generates about $\log m$ new vectors, where m is the number of blocks in the message to be encrypted.

As before the message P to be encrypted, is divided into blocks of length n each. Let these blocks be $P_1, P_2, ..., P_{m-1}$. The initial vector r is used to generate $t = \lceil \log(m+2) \rceil$ new vectors, which in turn are used to prepare $m + 1$ new pairwise independent vectors $S_0, S_1, ..., S_m$.

The following pseudo-code is the proposed method of generating the sequence S.

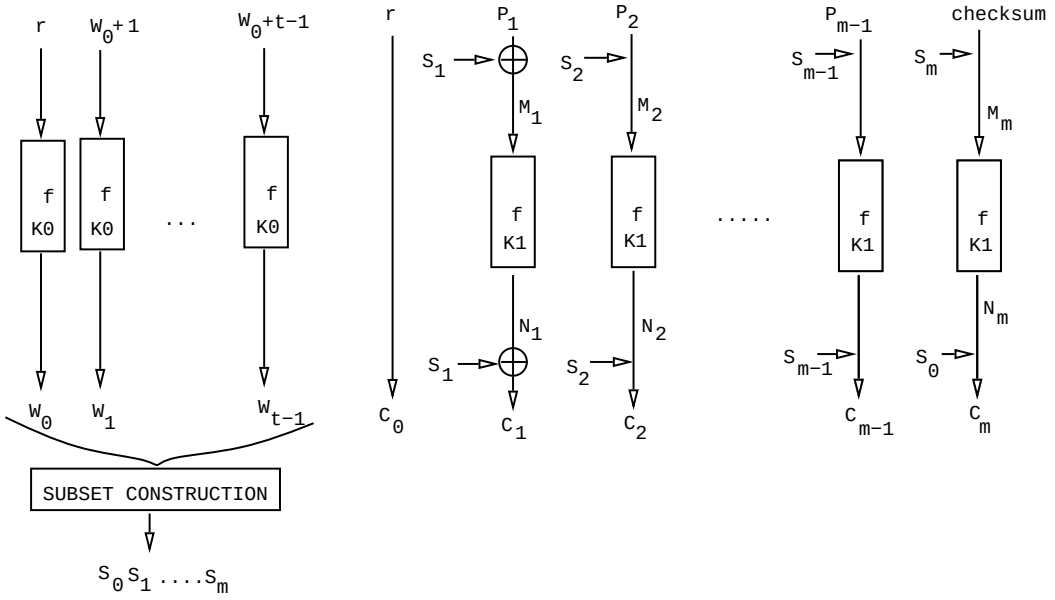

Figure 2: IAPM with only xor operations

procedure pairwise_independent_sequence(*in* $r, m, K0$; *out* S)
{
$W_0 = f_{K0}(r)$;
$S_0 = W_0$;
for $i = 1$ to m do
    $j = i + 1$;
    $k = 0$;
    /* find the index if the least significant ON bit in $(i + 1)$ */
    while $((j\&1) == 0)$ do
        $k = k + 1$; $j = j >> 1$; /* increment k and right shift */
    end while
    if $((j \oplus 1) == 0)$ /* if $(i + 1)$ is a power of 2 */
        $W_k = f_{K0}(W_0 + k)$;
    $S_i = S_{i-1} \oplus W_k$;
end for
}

Note that S_i is obtained from S_{i-1} in just one XOR. The inner while loop condition is checked two times on average.

The ciphertext message $C =< C_0, C_1, ..., C_m >$ is generated as follows (see fig 2):

$C_0 = r$
for $i = 1$ to $m - 1$ do
    $M_i = (P_i \oplus S_i)$
    $N_i = f_{K1}(M_i)$
    $C_i = (N_i \oplus S_i)$
end for
checksum $= P_1 \oplus P_2 \oplus ... \oplus P_{m-1}$
$M_m = (\text{checksum} \oplus S_m)$
$N_m = f_{K1}(M_m)$
$C_m = (N_m \oplus S_0)$

Again, note that S_0 is used in the last step. This pseudo-code is same as the one in the previous section except that all integer additions have been replaced by exclusive or operations.

Here is the pseudo-code for decryption:

$r = C_0$
invoke pairwise_independent_sequence$(r, m, K0, S)$;
for $i = 1$ to $m - 1$ do
    $N_i = (C_i \oplus S_i)$
    $M_i = f_{K1}^{-1}(N_i)$
    $P_i = (M_i \oplus S_i)$
end for
checksum $= P_1 \oplus P_2 \oplus ... \oplus P_{m-1}$
$N_m = (C_m \oplus S_0)$
$M_m = f_{K1}^{-1}(N_m)$
$P_m = (M_m \oplus S_m)$
Integrity $\equiv$ $(P_m ==$ checksum$)$

4. Performance

The IAPM scheme was implemented for DES on IBM PowerPC (200MHz). For messages of size 1024 64-bit blocks the IAPM scheme (with the prime number construction) yielded throughput of 34 Mbits/sec. In comparison, the CBC scheme (i.e. just encryption) ran at 35.5 Mbits/sec.

5. Patents

IBM has filed a U.S. patent on all these schemes.