Optimized Self-Synchronizing Mode of Operation

Ammar Alkassar[1], Alexander Geraldy[2], Birgit Pfitzmann[2], and Ahmad-Reza Sadeghi[2]

[1] Sirrix AG, D-66424 Homburg, Germany. alkassar@sirrix.com
[2] Universität des Saarlandes, FR 6.2 Informatik, D-66123 Saarbrücken, Germany.
ageraldy@krypt.cs.uni-sb.de, {pfitzmann, sadeghi}@cs.uni-sb.de

Abstract. Modes of operation adapt block ciphers to many applications. Among the encryption modes, only CFB (Cipher Feedback) has both of the following properties: Firstly it allows transmission units shorter than the block-cipher length to be encrypted and sent without delay and message expansion. Secondly, it can resynchronize after the loss of such transmission units.
However, CFB is inefficient in such applications, since for every transmission unit, regardless how short, a call to the block cipher is needed.
We propose a new mode of operation based on CFB which remedies this problem. Our proposal, OCFB, is almost optimally efficient (i.e., almost as many message bits are encrypted as block-cipher output bits produced) and it can self-synchronize after the loss or insertion of transmission units. We prove the security of CFB and OCFB in the sense of modern cryptography.

1 Introduction

Symmetric-key block ciphers are one of the most prominent building blocks in many cryptographic systems. They are used to construct various primitives such as stream ciphers, message authentication codes, and hash functions. Conceptually, a block cipher is a function that maps fixed-size l-bit plaintext blocks to l-bit ciphertext blocks (l is called the length of block cipher). The function is parametrized by a key k of a certain length. Examples of well-known block ciphers are Triple-DES (based on [DES77]), IDEA [LM91] and the AES candidates [NIST00], in particular Rijndael [DR99]. To encrypt longer messages and to fulfil varying application requirements, several modes of operation for block ciphers have been proposed. Among the standardized encryption modes from [FIPS81], CBC (Cipher Block Chaining) and CFB (Cipher Feedback) use the previous ciphertext block in the encryption so that each ciphertext block depends on the preceding plaintext, while OFB (Output Feedback) acts as pseudo-random bit generator and allows a large part of the encryption procedure to be precomputed. CFB and OFB can be used for applications where plaintext units with $L < l$ bits ($L = 8$ and $L = 1$ are typical cases) must be encrypted and

M. Matsui (Ed.): FSE 2001, LNCS 2355, pp. 78–91, 2002.

transmitted without delay. We call this a *transmission unit*. The counter mode (e.g., [DH79,LRW00]) can replace OFB.

Several applications need a *self-synchronizing* mode of operation, i.e., an error in the ciphertext must only lead to a small amount of incorrect plaintext. These are applications where the protocols have no or very basic built-in fault tolerance because the data type, e.g., voice or video, is such that small errors are unnoticeable or recoverable by natural redundancy. A more systematic alternative to a self-synchronizing cipher would be to add more error correction either to the transmission system or the application, but in practice, both may be fixed and one has to offer a transparent encryption layer in between. An application where we encountered this problem is ISDN, a common network for the subscriber area in Europe and Japan [Boc92], in particular for integrating phone, fax and Internet access. An effective way to secure the voice and fax communication, too, is transparent encryption directly before the ISDN network termination, e.g., by an ISDN card together with software encryption. Here one has precisely the network and application conditions for a self-synchronizing cipher: no underlying error correction, no message expansion possible (at least not without administrational problems), and the applications tolerate small errors but not desynchronization.

There are different types of errors. We speak of *bit errors* if certain bits are flipped, but the number of bits is unchanged. (Hence we include some burst errors in this class.) *Slips* are errors where bits are lost or inserted. Apart from outside disturbances, slips result from crossing networks with different clock frequency. CBC and CFB only tolerate slips if entire transmission units are inserted or lost. Hence only CFB can be used on networks where slips for units smaller than the block-cipher length occur. For instance, ISDN is byte-synchronous, i.e., slips can only be multiples of $L = 8$ bits [Boc92]. Hence CFB with 8-bit transmission units can be used. If nothing at all is known about the slips, $L = 1$ must be used. However, CFB is inefficient in such cases, since for every transmission unit the block cipher is called to encrypt l bits, e.g., 64 or 128. This is $n = l/L$ times as often as in other modes.

In this paper we present a solution OCFB to this problem. It is based on CFB, almost as efficient as CFB with $L = l$ and self-synchronizing even for $L < l$. We prove the concrete security of CFB and OCFB in the sense of [BDJR97]. To our knowledge this is also the first rigorous security analysis of CFB.

These properties make our proposal very appealing also to all applications which already use CFB.

Related literature: CFB and OCFB are so-called single modes of operation. Recently, multiple modes of operation have found considerable attention, but results as in [Bih94,Bih98,Wag98,HP99] suggest that single modes with a better block cipher are more promising. Now that an AES winner has been announced, this seems realistic.

Apart from that, recent research on modes of operation concentrates on MAC modes, e.g., [BR00,PR00,CKM00] and modes combining integrity and secrecy [GD00,Jut00,Rog00]. This is clearly important for many applications

and the natural way to build new applications, but as motivated above, well-established networks and applications remain where self-synchronization is important.

Proving the security of modes of operation started with [BKR94,BGR95], and a large part of the new literature cited above contains such proofs. Encryption modes (CBC and counter) were first treated in [BDJR97], and here also the concrete security of symmetric encryption was defined in detail.

The only fairly recent security analysis for CFB we know is [PNRB94], but rather from the point of view that CFB does not prevent differential and linear cryptanalysis of the block cipher.

Outline of the paper: We first present some basics and notations: the transmission model, notation for encryption, and CFB and its self-synchronization properties. After this we present our proposed operation mode OCFB and its self-synchronization property and compare the efficiency of CFB and OCFB. Finally, we present left-or-right security [BDJR97] as a basis for our security treatment and give concrete security proofs for both standard CFB and OCFB.

2 Preliminaries

2.1 Transmission Model

As motivated above, we assume a communication system with L-bit transmission units. The most important fact is that slips, i.e., losses or insertions, only occur for entire transmission units. The main case we consider is $L = 8$. We assume that we are not allowed any message expansion.

2.2 Encryption Notation

A block cipher is a triple (gen, enc, dec). The algorithm gen randomly generates a key k, which is used by the encryption function $enc_k : \{0,1\}^l \to \{0,1\}^l$ to encrypt a plaintext message m to a ciphertext $c = enc_k(m)$. The decryption function dec_k decrypts a ciphertext c' to $m' = dec_k(c')$ using the same key k. The value l is called the block-cipher length.[1]

CFB and OCFB use l-bit shift registers consisting of $n = l/L$ positions $SR[1], \ldots, SR[n]$ of L-bit transmission units. By $r = SR$ or $SR = r$ we denote reading or writing the entire register. Shifting a transmission unit m into the register is written $SR \leftarrow m$, i.e., this means $SR[i] = SR[i+1]$ for $i = 1, \ldots, n-1$ and $SR[n] = m$. Shifting a transmission unit out is written $m \leftarrow SR$, i.e., this means $m = SR[1]$, $SR[i] = SR[i+1]$ for $i = 1, \ldots, n-1$ and $SR[n] = 0$.

Other notations are $|m|$ for the length of a string and $\oplus$ for the bitwise exor of strings of equal length. By $a \in_R S$ we denote the random and uniform selection of an element a from the set S. Further, $P(pred(x) :: x = A(z))$ represents the probability that x fulfils a certain predicate $pred$ if x is chosen with the probabilistic algorithm A on input z.

[1] CFB and OCFB do not use dec_k or the fact that enc_k is a permutation; hence we could also define them with an arbitrary family F of functions enc_k.

2.3 Cipher Feedback Mode

The *Cipher Feedback Mode* (CFB) is illustrated in Figure 1. More precisely, $CFB^{l,L}$ denotes the version with an l-bit block cipher and L-bit transmission units, where $l = Ln$.

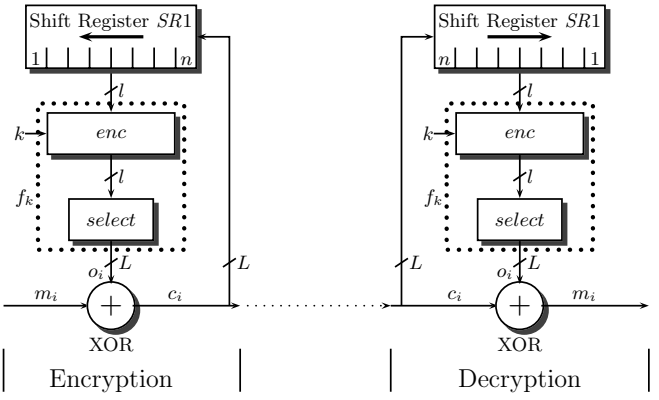

Fig. 1. Cipher Feedback Mode

In the figure, the block cipher enc_k followed by a fixed selection of L output bits is summarized as a function $f_k : \{0,1\}^l \to \{0,1\}^L$. The values m_i and c_i denote plaintext and ciphertext transmission units. A shift register $SR1$ holds the last n ciphertext transmission units, i.e., $SR1_i = c_{i-n} \dots c_{i-1}$ in Round $i > n$. In round $i = 1$, it contains an initialization vector $SR1_1 = IV$. Then the encryption algorithm of $CFB^{l,L}$ for the i-th transmission unit is:

1. $o_i = select(enc_k(SR1_i))$;
2. $c_i = m_i \oplus o_i$;
3. $SR1_{i+1} = (SR1_i \leftarrow c_i)$.

For decryption, Step 2 is replaced by $m_i = c_i \oplus o_i$.

Error Propagation: Under the given error model, i.e., if slips only occur for multiples of transmission units, errors disturb decryption only as long as they remain in the shift register of the receiver.

3 Optimized Cipher Feedback (OCFB)

For communication systems with $L \ll l$, CFB is inefficient. For instance, for $L = 8$, it calls enc_k for every single transmitted byte. This is less efficient than other modes by a factor of $n = l/L$, e.g., 8 for DES and 16 for AES.

The efficiency of CFB can be optimized by buffering all l output bits of enc_k and using them for successive transmission units, using a counter to trigger a

call of enc_k every n-th transmission unit. However, this would destroy the self-synchronization for slips of individual transmission units because the counters of the sender and the recipient would lose their synchrony (relative to the ciphertext stream). Hence the counters must be resynchronized. As the transmission model does not allow message expansion, this can only be done via the ciphertext itself. The idea is to use a *synchronization pattern*; this is sketched in Figure 2. Each automaton (for encryption and decryption) compares the content of its shift register $SR1$ with this pattern after each transmission unit. If it finds the pattern, it resets its counter; this reset synchronizes the counters of the two automata.

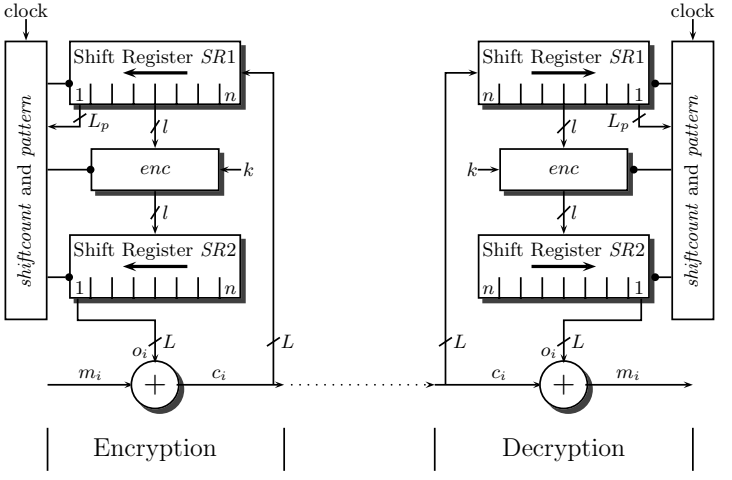

Fig. 2. OCFB

Let *pattern* denote the pattern, L_P its length, and p the probability that it matches a random register content. The pattern can be fixed at an arbitrary time, it need not be secret or random. The counter is called *shiftcount*, and *match* denotes the pattern-matching algorithm, typically a simple string comparison, possibly with wildcards. One can start with fixed initialization vectors in the registers $SR1$ and $shiftcount = n - 1$, but also asynchronously. Then the encryption algorithm of $OCFB^{l,L,p}$ for the i-th transmission unit is:

1. If $match(SR1_i, pattern)$ then $shiftcount = n$ else $shiftcount{+}{+}$;
2. if $shiftcount = n$ then $SR2_i = enc_k(SR1_i)$; $shiftcount = 0$;
3. $SR2_{i+1} = (o_i \leftarrow SR2_i)$;[2]
4. $c_i = m_i \oplus o_i$;
5. $SR1_{i+1} = (SR1_i \leftarrow c_i)$.

[2] If encryption is called in the next round, two values are called $SR2_{i+1}$. When in doubt, the second (final) one is meant.

For decryption, Step 4 is replaced by $m_i = c_i \oplus o_i$.

CFB can be seen as a special case of OCFB where the synchronization pattern is found in every round (e.g., by using the pattern length zero).

4 Efficiency of CFB and OCFB

We define the efficiency eff_M of a mode of operation M as the average number of encrypted plaintext bits per call to the l-bit block cipher, divided by l. The optimum with any normal mode of operation is therefore 1.The efficiency of $CFB^{l,L}$ is $\mathit{eff}_{CFB^{l,L}} = L/l$. The efficiency of OCFB is $E(X)/n$, where X, called *distance*, is the random variable describing the number of transmission units between two calls to the block cipher, and $E(X)$ its expectation. Recall that the pattern matches random data with probability p.Let $\overline{p} = 1 - p$, and for the pattern length we assume $L_p \leq L$.[3,4]

If $\mathit{shiftcount} = n$, the block cipher is always called. Hence, assuming that the values c_i are random (this will be justified in Section 5), the distance X has a cut-off geometric distribution: $P(X = j) = \overline{p}^{j-1}p$ for $0 < j < n$ and $P(X = n) = \overline{p}^{n-1}$. Hence

$$E(X) = \sum_{j=1}^{n} jP(X = j) = \sum_{j=1}^{n-1} jp\overline{p}^{j-1} + n\overline{p}^{n-1}$$
$$= \frac{1 - \overline{p}^n}{1 - \overline{p}}.$$

This can be verified by multiplying the terms out; a similar formula can, e.g., be found in [Nel95].Hence the efficiency is $\mathit{eff}_{OCFB^{l,L,p}} = (1 - \overline{p}^n)/(np)$.
For instance, for $l = 64$ and $L = 8$, an 8-bit pattern with $p = 2^{-8}$ gives an efficiency of 0.986, i.e., one block-cipher call for 7.89 bytes on average.

Self-synchronization for bit errors is as in CFB. Self-synchronization after a slip takes somewhat longer, $1/p$ transmission units on average. Tests have shown that this is no problem for phone and fax with the parameters from the previous example and the usual error rates of ISDN. Even with much larger p, e.g., $p = 1/n$ so that resynchronization after slips happens after about one l-bit block just as for bit errors, the efficiency of OCFB is still far superior to that of CFB. E.g., for $l = 64$ and $L = 8$, we then get $\mathit{eff}_{OCFB^{l,L,p}} = 1-(1-1/n)^n \approx 0.66$, which is more than 5 times faster than CFB.

[3] For our main example $L = 8$, this is reasonable and simplifies the analysis because successive matchings are independent. For small L (in particular, $L = 1$) we advise a pattern like 100...000 where repetition is only possible after L_p/L transmission units.

[4] Depending on the soft- or hardware configuration, it may be useful to buffer the values o_i to avoid problems if the pattern sometimes repeats faster than its expectation. In the definition, Step 4 becomes $c_i = m_i \oplus o_{i-\beta}$ for a buffer length β. The computation of the o_i's and the exors are then almost asynchronous. The security analysis is easily adapted to this case.

5 Security

In this section we prove the concrete security of CFB and OCFB, using the first proof as a basis for the second one. The proof follows the pattern of other security proofs of modes of operation: The block cipher is modeled as a pseudo-random function [GGM86] or rather, for concrete security, one parametrizes the effort an adversary needs to notice non-random properties [BKR94]. One then shows that any attack against the mode of operation that succeeds with a certain probability given a certain amount of resources would give an attack on the underlying block cipher, again with precise resources and probability.

5.1 Left-or-Right Security

Left-or-right security was introduced in [BDJR97] as a strong (adaptive) form of chosen-plaintext security.[5] The attack is modeled as a game between an active adversary (*left-right distinguisher*) D_{lr} and an encryption oracle $\mathcal{E}_{k,b}$, which contains a key k and a bit $b \in \{0,1\}$. In each round, D_{lr} chooses two plaintexts m_i^0 and m_i^1 with $|m_i^0| = |m_i^1|$ and gives them to $\mathcal{E}_{k,b}$. This oracle returns $c_i = enc_k(m_i^b)$. (The cases $b = 0$ and $b = 1$ are called left and right case.) Finally, D_{lr} outputs a bit e, meant as a guess at b. The adversary's advantage $Adv_{D_{lr}}$ is the defined as for a statistical test, i.e., the probability difference of the output $e = 0$ in the two cases.

The adversary's resources are parametrized by its maximum running time t, the number of queries q to the encryption oracle, and their total length μ. Its maximum success probability is called ε.

Definition 1 (Left-or-right security [BDJR97]). *An encryption scheme (gen, enc, dec) is (t, q, μ, ε)-secure in the left-or-right sense if for any adversary D_{lr} which runs in time at most t and asks at most q queries, these totaling at most μ bits,*

$$Adv_{D_{lr}} = P[{D_{lr}}^{\mathcal{E}_{k,0}} = 0 :: k \leftarrow gen] - P[{D_{lr}}^{\mathcal{E}_{k,1}} = 0 :: k \leftarrow gen] \leq \varepsilon.$$

The first term describes the probability that D_{lr} outputs $e = 0$ when interacting with the oracle containing $b = 0$, and the second term the corresponding probability for an oracle with $b = 1$. The definition is illustrated for CFB in Figure 3.

For modes of operation, the messages m_i^b that the adversary can choose adaptively may be individual transmission units.

Chosen-ciphertext security is not required in [BDJR97], and it cannot be required in the strict sense for a self-synchronizing cipher: This would correspond to non-malleability, but the purpose of self-synchronization is precisely to make slightly distorted ciphertexts decrypt to related cleartexts.

[5] More precisely, [BDJR97] contains four definitions and relations among them. It is shown that concrete encryption systems are best proven with respect to left-or-right security, since this implies good reductions to the other definitions.

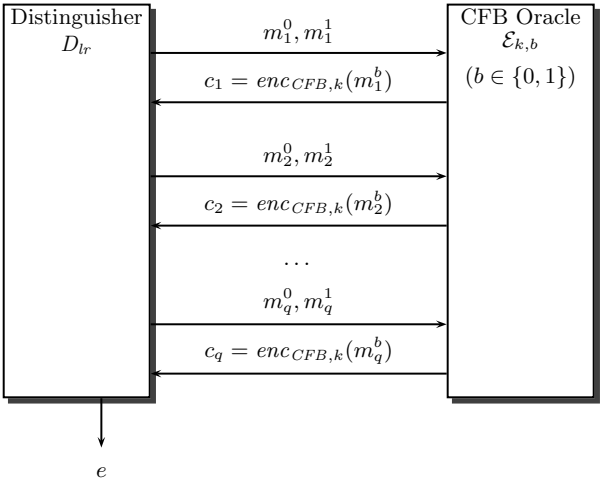

Fig. 3. Left-or-right security

5.2 Function Families

When considering a mode of operation, the block cipher is typically modeled as a pseudo-random function. In the case of CFB a weaker assumption is sufficient: Only the family of functions $f_k = select(enc_k(\cdot))$ must be pseudo-random (see Figure 1).

We use the concrete-security definitions from [BKR94,BDJR97]. An (l, λ)-*function family* is a multiset F of functions $f_k : \{0,1\}^l \to \{0,1\}^\lambda$. Alternatively, one writes $f \in_R F$ for the random choice of a function from the family, or $k \leftarrow gen$ if an algorithm is given that generates a key such that f_k is random in F.

The (l, λ)-*random function family* $R^{l,\lambda}$ consists of *all* functions $rf : \{0,1\}^l \to \{0,1\}^\lambda$. A key is simply an entire function rf. Clearly, such a key is much too long in practice, but pseudo-randomness of other (l, λ)-function families F is defined relative to $R^{l,\lambda}$: An adversary D_{prf} interacts with an oracle $\mathcal{F}$ that has chosen a function f randomly from either F or $R^{l,\lambda}$. In each round, D_{prf} may send a value $x \in \{0,1\}^l$ and $\mathcal{F}$ answers with $f(x)$. Again, the adversary has to output a bit meant as a guess at the function family, and its advantage is defined as for a statistical test.

Definition 2 (Pseudo-random functions [BDJR97]). *An (l, λ)-function family F is a (t, q, ε)-secure PRF family if for any distinguisher D_{prf} making at most q oracle queries and running in time at most t,*

$$Adv_{D_{prf}}(F) = P[{D_{prf}}^f = 0 :: f \in_R F] - P[{D_{prf}}^f = 0 :: f \in_R R^{l,\lambda}] \leq \varepsilon.$$

By $CFB^{l,L}(F)$ we denote CFB used with a certain function family F in the place of the functions $f_k = select(enc_k(\cdot))$. Similarly, $OCFB^{l,L,p}(F)$ denotes OCFB with F as the functions enc_k.

5.3 Security with Random Functions

Before proving a certain degree of concrete security, it is useful to consider what one can reasonably expect. This is a birthday bound, similar to other modes with feedback. The basic idea is that left-or-right security breaks down at the first repetition of the value of $SR1$: The adversary D_{lr} knows the values in $SR1$ (they are ciphertexts, at least after the initialization vector). If $SR1_i = SR1_j$ for $i \neq j$, then also $o_i = o_j$ in CFB. Hence $c_i \oplus c_j = m_i^b \oplus m_j^b$, and thus b is revealed if $m_i^0 \oplus m_j^0 \neq m_i^1 \oplus m_j^1$, which will typically be the case. For OCFB, a similar argument holds, but only those values in $SR1$ must be considered where encryption is called, i.e., on average they are fewer by a factor of $n \cdot \mathit{eff}_{OCFB}$.

We now show that if CFB were used with real random functions, there would indeed be no better attack than waiting for collisions among values of $SR1$, and we derive the resulting concrete security.

Lemma 1 (Concrete security of CFB with random functions). *CFB with random functions, i.e., $CFB^{l,L}(R^{l,L})$, is $(t, q, Lq, \varepsilon_{CFB^{l,L},q})$-secure in the left-or-right sense for any t (i.e., no runtime restriction), any number q of queries (of one transmission unit each), and $\varepsilon_{CFB^{l,L},q} = q(q-1)2^{-l-1}$.*

Proof. We abbreviate probabilities in the left-or-right game with bit b as P_b, and for intermediate values in such a game (e.g., c_i) we use the notation introduced in the text (see also Figure 3).For instance, the advantage can then be written $Adv_{D_{lr}} = P_0[e = 0] - P_1[e = 0]$.

We distinguish whether a collision occurs during the attack or not. Let C be the *collision event*, i.e., it contains all runs of the game where $i \neq j$ exist with $1 \leq i, j \leq q$ and $SR1_i = SR1_j$. Its complement is called $\overline{C}$.

As long as no collision has occurred, each value $o_i = f(SR1_i)$ can be seen as randomly and independently chosen (by the definition of random functions), i.e., it is a one-time pad for the plaintext transmission unit m_i^b. Hence c_i is random and independent of $c_1, \ldots, c_{i-1}$ and $m_1^b, \ldots, m_i^b$. Thus the collision probability in round i does not depend on b, and overall we can abbreviate

$$P[C] = P_0[C] = P_1[C]. \tag{1}$$

For the same reason, collisions are the only help for the adversary. If no collision occurs, the adversary outputs $e = 0$ with the same probability for $b = 0$ and $b = 1$.

$$P_0[e = 0 \mid \overline{C}] \;=\; P_1[e = 0 \mid \overline{C}]. \tag{2}$$

We can therefore rewrite the adversary's advantage as follows (using (1) and (2) in the last equality):

$$\begin{aligned} Adv_{D_{lr}} &= P_0[e=0] - P_1[e=0] \\ &= P_0[e=0 \mid C]\ P_0[C] + P_0[e=0 \mid \overline{C}]\ P_0[\overline{C}] \\ &\quad - P_1[e=0 \mid C]\ P_1[C] - P_1[e=0 \mid \overline{C}]\ P_1[\overline{C}] \\ &= P[C](P_0[e=0 \mid C] - P_1[e=0 \mid C]) \\ &\leq P[C]. \end{aligned}$$

Collisions: For the collision probability, we cannot simply use the birthday formula because $SR1_i$ and $SR1_j$ are not independent if $|j-i| < n$. We then say that $SR1_i$ and $SR1_j$ *overlap*.

We define the stream $B = IV\ c_1 \dots c_{q-1}$ of all the collision-relevant transmission units, i.e., those shifted through $SR1$ until the last, q-th, encryption. The length of B is $Q = (n+q-1)L$ bits, and the shift register contents are $SR1_i = B[i] \dots B[i+n-1]$ for $i = 1, \dots, q$. We first derive the number $col_{i,j}$ of streams with a collision $SR1_i = SR1_j$ for every possible pair (i,j), i.e., $1 \leq i < j \leq q$. We do this for the case with random initialization vector IV. (The difference in the other case is negligible).

a) Without overlapping, i.e., $j \geq i+n$:
   As $SR1_i = SR1_j$, there are 2^l possible values for the shift register contents of both rounds. The remaining $Q-2l$ bits offer 2^{Q-2l} possibilities. Thus $col_{i,j} = 2^{Q-l}$.
b) With overlapping, i.e., $i < j < i+n$:
   Let $d = j-i$. Then $SR1_i$ and $SR1_j$ together use $l+dL$ bits, and those have 2^{dL} possible values because $SR1_i[1] = SR1_i[1+d], \dots, SR1_i[n-d] = SR1_i[n]$. The remaining $Q-l-dL$ bits offer 2^{Q-l-dL} possibilities. Thus again $col_{i,j} = 2^{dL} \cdot 2^{Q-l-dL} = 2^{Q-l}$.

There are $q(q-1)/2$ possible pairs (i,j). Hence the number col of streams B with at least one collision is less than $q(q-1)2^{Q-l-1}$. (In the bound, streams with several collisions are counted several times.)

We have shown above that each stream *without* collision has the probability 2^{-Q} (because each new c_i is uniformly distributed). Hence $P[\overline{C}] = (2^Q - col)/2^Q > 1 - q(q-1)2^{-l-1}$. This implies

$$P[C] \leq q(q-1)2^{-l-1},$$

which remained to be shown. □

Lemma 2 (Concrete security of OCFB with random function).
For all parameters, OCFB with random functions is at least as secure as CFB for the same length of the block cipher and transmission units, i.e., $OCFB^{l,L,p}(R^{l,l})$ is at least as secure as $CFB^{l,L}(R^{l,L})$.

Proof. The structure of the proof is similar to that for CFB.

First we have to show again that each value o_i, and thus also c_i, is random and independent of $c_1, \ldots, c_{i-1}$ and $m_1^b, \ldots, m_i^b$. By the use of *shiftcount* in the encryption algorithm, $SR2$ is set to a value $rf(SR1_i)$ at least every n transmission units, where rf is the random function replacing enc_k. This gives n potential values $o_i, \ldots, o_{i+n-1}$ which are totally random at this point in time. The adversary may choose $m_{i+1}^b, \ldots, m_{i+n-1}^b$ adaptively after this, but no information at all about o_{i+j} is given out before its use in c_{i+j}. Thus the claimed independence holds. This implies, as in the proof of Lemma 1, that $Adv_{D_{lr}} \leq P[C_{enc}]$, where C_{enc} is the event that there is a collision $SR1_i = SR1_j$ for two indices i, j where encryption (i.e., here rf) is called.

Clearly $P[C_{enc}] < P[C]$, the probability of collisions for arbitrary indices i, j. Hence, with the results of the previous paragraph, an upper bound for $P[C]$ can be computed literally as for CFB. □

This lemma shows that $OCFB^{l,L,p}(R^{l,l})$ is $(t, q, Lq, \varepsilon_{CFB^{l,L},q})$-secure for all t and q. In fact, the security is even somewhat better: It is approximately $(t, q, Lq, \varepsilon_{OCFB^{l,L,p},q})$, where $\varepsilon_{OCFB^{l,L,p},q} = q'(q'-1)2^{-l-1}$ with $q' = q/(n \cdot \mathit{eff}_{OCFB^{l,L,p}})$. This follows by counting only the collisions in C_{enc} in the second part of the proof: On average, encryption is called for every $(n \cdot \mathit{eff}_{OCFB^{l,L,p}})$-th index. Thus q' roughly plays the role of q in the proof above, i.e., the number of possibilities for i and j.

5.4 Security with Pseudo-Random Functions

Now we want to prove the security of CFB and OCFB with pseudo-random functions, i.e., in the real world, as far as the block cipher is pseudo-random.

Theorem 1 (Concrete security of CFB with pseudo-random functions).
Let F be a (t', q', ε')-secure (l, L)-bit pseudo-random function family, and t_{CFB} the time needed for one CFB round without the computation of f_k. Then $CFB^{l,L}(F)$ is (t, q, μ, ε)-secure in the left-or-right sense with $q = q'$, $\mu = q'L$, $t = t' - qt_{CFB} - t_{const}$ for a small constant t_{const}, and $\varepsilon = 2\varepsilon' + \varepsilon_{CFB^{l,L},q}$.

These bounds are very good: The allowed time t for an attack is almost t' (because t' is the time for an attack on F, while qt_{CFB} is time needed for the correct use of CFB), and an addition of $2\varepsilon'$ is standard for this kind of proof (see [BDJR97]).

The basic idea of the proof is the standard pseudo-randomness argument: If an adversary could break left-or-right security of CFB with a pseudo-random function family better than this is possible with random functions, one could use this adversary as a distinguisher between real and pseudo-random functions. The following reduction gives the concrete security for this argument.

Proof. We assume that a distinguisher D_{lr} contradicts the theorem. We construct a distinguisher D_{prf} which contains D_{lr} as a black box, playing the left-or-right

game with a CFB oracle, see Figure 4. The CFB oracle is implemented by D_{prf} itself, using its own oracle $\mathcal{F}$ (containing a random or pseudo-random function) as f_k. Hence D_{prf} chooses b itself (randomly) and can compare D_{lr}'s output e with b. It outputs the Boolean value $e^* = \neg(e = b)$.

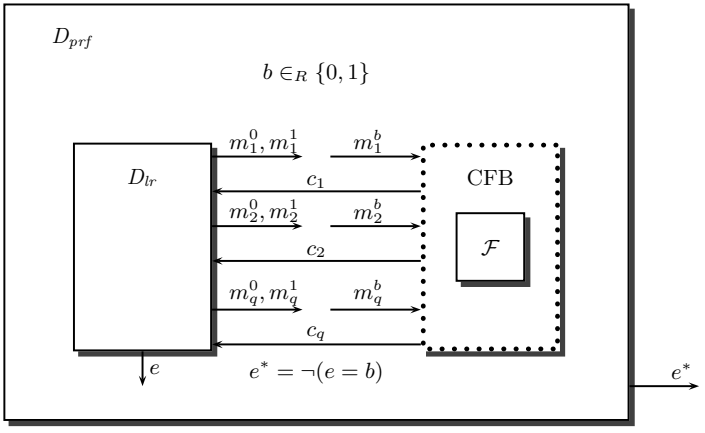

Fig. 4. Reduction of CFB to PRF

For each query of D_{lr} to the assumed CFB oracle, D_{prf} must make one query to $\mathcal{F}$ and update the CFB state. Hence it makes $q' = q$ queries and takes time $t' = t + qt_{CFB} + t_{const}$, where t_{const} is the time for set-up and the final comparison. Hence D_{prf} is a distinguisher against which the function family F is assumed to be secure.

We now abbreviate $R^{l,L}$ by R, and for $G \in \{R, F\}$ we write $Adv_{D_{lr}}(G)$ for the advantage of D_{lr} (according to Definition 2) when $CFB^{l,L}(G)$ is used. Furthermore, let P_F and P_R denote the two probability spaces from Definition 2. Then we can write the advantage of D_{prf} as

$$Adv_{D_{prf}} = P_F[e^* = 0] - P_R[e^* = 0],$$

and for $G \in \{R, F\}$, we can continue

$$\begin{aligned} P_G[e^* = 0] &= P[e^* = 0 :: f \in_R G; e^* \leftarrow {D_{prf}}^f] \\ &= P[e = b :: f \in_R G; b \in_R \{0,1\}; e \leftarrow {D_{lr}}^{\mathcal{CFB}_{f,b}}]. \end{aligned}$$

Here we used the construction of D_{prf} and introduced the notation $\mathcal{CFB}_{f,b}$ (corresponding to Definition 1) for a CFB left-or-right oracle with the particular function f and bit b. As the choices of f and b are independent, b can be chosen first and we get

$$P_G[e^* = 0] = \frac{1}{2} \sum_{b \in \{0,1\}} P[D_{lr}{}^{\mathcal{CFB}_{f,b}} = b :: f \in_R G]$$
$$= \frac{1}{2}(P[D_{lr}{}^{\mathcal{CFB}_{f,0}} = 0 :: f \in_R G] + 1 - P[D_{lr}{}^{\mathcal{CFB}_{f,1}} = 0 :: f \in_R G])$$
$$= \frac{1}{2} + \frac{1}{2} Adv_{D_{lr}}(G).$$

We have assumed $Adv_{D_{lr}}(F) > \varepsilon$, and by Lemma 1, $Adv_{D_{lr}}(R) \leq \varepsilon_{CFB^{l,L},q}$. Hence

$$Adv_{D_{prf}} = \frac{1}{2}(Adv_{D_{lr}}(F) - Adv_{D_{lr}}(R)) > \frac{1}{2}(\varepsilon - \varepsilon_{CFB^{l,L},q}) = \varepsilon'.$$

This is the desired contradiction to the security of the pseudo-random function family F. □

Theorem 2 (Concrete security of OCFB with pseudo-random functions).
OCFB is as secure as CFB in the following sense: Let F be a (t', q', ε')-secure (l, l)-bit pseudo-random function family. Then $OCFB^{l,L,p}(F)$ is (t, q, μ, ε)-secure in the left-or-right sense with $q = q'$, $\mu = qL$, $t = t' - qt_{OCFB} - t_{const}$, and $\varepsilon = 2\varepsilon' + \varepsilon_{CFB^{l,L},q}$.

Proof. This proof is almost identical to that for CFB. The only difference is that D_{lr} simulates an OCFB-oracle instead of a CFB-oracle, and therefore the time for each call is t_{OCFB} instead of t_{CFB}. □

In fact, the security is better: On average, D_{lr} only needs $q' = q/(n \cdot \mathit{eff}_{OCFB^{l,L,p}})$ calls to its own oracle $\mathcal{F}$ (corresponding to the block cipher) to encrypt the q transmission units, i.e., q can be chosen correspondingly larger.

Furthermore, as discussed after Lemma 2, the bound $\varepsilon_{CFB^{l,L},q}$ in that lemma is not quite tight. The proof of Theorem 2 remains unchanged for whatever value $\varepsilon_{OCFB^{l,L,p},q}$ is proven in Lemma 1.

Acknowledgment. We thank Michael Braun, Jörg Friedrich, Matthias Schunter, and Michael Waidner for interesting discussions and helpful comments.

References

[BDJR97] M. Bellare, A. Desai, E. Jokipii, P. Rogaway: A Concrete Security Treatment of Symmetric Encryption: Analysis of the DES Modes of Operation, 38th Symposium on Foundations of Computer Science (FOCS), IEEE Computer Society, 1997, 394–403.

[BGR95] M. Bellare, R. Guérin, P. Rogaway: XOR MACs: New Methods for Message Authentication Using Finite Pseudorandom Functions; Crypto '95, LNCS 963, Springer-Verlag, Berlin 1995, 15–28.

[Bih94] E. Biham: On Modes of Operation; Fast Software Encryption '93, LNCS 809, Springer-Verlag, Berlin 1994, 116–120.

[Bih98] E. Biham: Cryptanalysis of Multiple Modes of Operation; Journal of Cryptography 11/1 (1998) 45–58.

[BKR94] M. Bellare, J. Kilian, P. Rogaway: The security of cipher block chaining; Crypto '94, LNCS 839, Springer-Verlag, Berlin 1994, 341–358.

[Boc92] P. Bocker: ISDN – The Integrated Services Digital Network; (2nd ed.), Springer-Verlag, Berlin 1992.

[BR00] J. Black, P. Rogaway: CBC macs for arbitrary-length messages: the three-key constructions; Crypto 2000, LNCS 1880, Springer-Verlag, Berlin 2000, 197–215.

[CKM00] D. Coppersmith, L. R. Knudsen, C. J. Mitchell: Key recovery and forgery attacks on the MacDES MAC algorithm; Crypto 2000, LNCS 1880, Springer-Verlag, Berlin 2000, 184–196.

[DES77] Specification for the Data Encryption Standard; Federal Information Processing Standards Publication 46 (FIPS PUB 46), 1977.

[DH79] W. Diffie, M. E. Hellman: Privacy and Authentication: An Introduction to Cryptography; Proceedings of the IEEE 67/3 (1979) 397–427.

[DR99] J. Daemen, V. Rijmen: The Rijndael Block Cipher, AES Proposal; `http://csrc.nist.gov/encryption/aes/rijndael/Rijndael.pdf,1999`.

[FIPS81] DES Modes of Operation; Federal Information Processing Standards Publication 81 (FIPS PUB 81) December 2, 1980.

[GD00] V. Gligor, P. Donescu: Integrity-Aware PCBC Encryption Schemes; Security Protocols 1999, LNCS 1796, Springer-Verlag, Berlin 2000, 153–171.

[GGM86] O. Goldreich, S. Goldwasser, S. Micali: How to construct random functions. Journal of the ACM 33/4 (1986) 210–217.

[HP99] H. Handschuh, B. Preneel: On the Security of Double and 2-Key Triple Modes of Operation; 6th International Workshop on Fast Software Encryption, LNCS 1636, Springer-Verlag, Berlin 1999, 215–230.

[Jut00] C. S. Jutla: Encryption Modes with Almost Free Message Integrity; NIST Workshop Symmetric Key Block Cipher Modes of Operation, Baltimore, October 2000, `http://csrc.nist.gov/encryption/aes/modes/`.

[LM91] X. Lai, J. Massey: A Proposal for a New Block Encryption Standard; Eurocrypt '90, LNCS 473, Springer-Verlag, Berlin 1991, 389–404.

[LRW00] H. Lipmaa, P. Rogaway, D. Wagner: CTR-Mode Encryption; NIST Workshop Symmetric Key Block Cipher Modes of Operation, Baltimore, October 2000, `http://csrc.nist.gov/encryption/aes/modes/`.

[Nel95] Randolph Nelson: Probability, Stochastic Processes, and Queuing Theory, Springer 95

[NIST00] National Institute of Standards (NIST): AES — Advanced Encryption Standard (AES) Development Effort; `http://csrc.nist.gov/encryption/aes/`, 1997–2000.

[PNRB94] B. Preneel, M. Nuttin, V. Rijmen, J. Buelens: Cryptanalysis of DES in the CFB mode; Crypto '93, LNCS 773, Springer-Verlag, Berlin 1994, 212–223.

[PR00] E. Petrank, C. Rackoff: CBC MAC for Real-Time Data Sources; Journal of Cryptology 13/3 (2000) 315–338.

[Rog00] P. Rogaway: OCB Mode: Parallelizable Authenticated Encryption; NIST Workshop Symmetric Key Block Cipher Modes of Operation, Baltimore, October 2000, `http://csrc.nist.gov/encryption/aes/modes/`.

[Wag98] D. Wagner: Cryptanalysis of some Recently-Proposed Multiple Modes of Operation; Fast Software Encryption '98, LNCS 1372, Springer-Verlag, Berlin 1998, 254–269.