New proofs for old modes

Mark Wooding
mdw@distorted.org.uk

13 March 2008

Abstract We study the standard block cipher modes of operation: CBC, CFB, and OFB and analyse their security. We don't look at ECB other than briefly to note its insecurity, and we have no new results on counter mode. Our results improve over those previously published in that (a) our bounds are better, (b) our proofs are shorter and easier, (c) the proofs correct errors we discovered in previous work, or some combination of these. We provide a new security notion for symmetric encryption which turns out to be rather useful when analysing block cipher modes. Finally, we pay attention to different methods for selecting initialization vectors for the block cipher modes, and prove security for a number of different selection policies. In particular, we introduce the concept of a 'generalized counter', and prove that generalized counters suffice for security in (full-width) CFB and OFB modes and that generalized counters encrypted using the block cipher (with the same key) suffice for all three modes.

Contents

- **1 Introduction** — 3
  - 1.1 Block cipher modes — 3
  - 1.2 Previous work — 3
  - 1.3 Our contribution — 3
  - 1.4 The rest of the paper — 4
- **2 Notation and definitions** — 4
  - 2.1 Bit strings — 4
  - 2.2 Other notation — 5
  - 2.3 Algorithm descriptions — 5
  - 2.4 Pseudorandom functions and permutations — 5
  - 2.5 Symmetric encryption — 6
  - 2.6 Initialization vectors and encryption modes — 9
- **3 Ciphertext block chaining (CBC) encryption** — 13
  - 3.1 Description — 13
  - 3.2 Security of CBC mode — 14
  - 3.3 Ciphertext stealing — 15
  - 3.4 Proof of theorem 3.2.1 — 16
- **4 Ciphertext feedback (CFB) encryption** — 19
  - 4.1 Description — 19
  - 4.2 Sliding strings — 20
  - 4.3 Security of CFB mode — 21
  - 4.4 Proof of theorem 4.3.1 — 22
- **5 OFB mode encryption** — 26
  - 5.1 Description — 26
  - 5.2 Security of OFB mode — 27
- **6 Acknowledgements** — 28
- **7 References** — 28

List of figures

- 3.1 Encryption using CBC mode — 13
- 3.2 Encryption and decryption using CBC mode with ciphertext stealing — 15
- 3.3 Garbage emitter W for CBC mode — 17
- 3.4 Notation for the proof of theorem 3.2.1. — 18
- 4.1 Encryption using CFB mode — 20
- 4.2 Adversary S attacking $\mathcal{E}\text{-}CFBL^{\mathcal{F}^{\ell,t},0^\ell}$ — 21
- 4.3 Notation for the proof of lemma 4.4.2. — 24
- 5.1 Encryption using OFB mode — 26

List of tables

- 1.1 Summary of our results — 4

1 Introduction

1.1 Block cipher modes

Block ciphers – keyed pseudorandom permutations – are essential cryptographic tools, widely used for bulk data encryption and to an increasing extent for message authentication. Because the efficient block ciphers we have operate on fixed and relatively small strings of bits – 64 or 128 bits at a time, one needs a 'mode of operation' to explain how to process longer messages.

A collection of encryption modes, named ECB, CBC, CFB and OFB, were defined in [Uni81]. Of these, ECB – simply divide the message into blocks and process them independently with the block cipher – is just insecure and not to be recommended for anything much. We describe the other three, and analyse their security using the standard quantitative provable-security approach. All three require an 'initialization vector' or 'IV' which diversifies the output making it hard to correlate ciphertexts with plaintexts. We investigate which conditions on these IVs suffice for secure encryption.

1.2 Previous work

The first quantitative security proof for a block cipher mode is the analysis of CBCMAC of [BKR94]. Security proofs for the encryption modes CBC and CTR appeared in [BDJR97], which also defines and relates the standard security notions of symmetric encryption. The authors of [AGPS01] offer a proof of CFB mode, though it is incomplete, as we discuss below.

1.3 Our contribution

We introduce a new security notion for symmetric encryption, named 'result-or-garbage', or 'ROG-CPA', which generalizes the 'real-or-random' notion of [BDJR97] and the 'random-string' notion of [RBBK01]. Put simply, it states that an encryption scheme is secure if an adversary has difficulty distinguishing true ciphertexts from strings chosen by an algorithm which is given only the *length* of the adversary's plaintext. This turns out to be just the right tool for analysing our encryption modes. We relate this notion to the standard 'left-or-right' notion and, thereby, all the others.

Our bound for CBC mode improves over the 'tight' bound proven in [BDJR97] by almost a factor of two. The difference comes because they analyse the construction as if it were built from a PRF and add in a 'PRP-used-as-a-PRF' correction term: our analysis considers the effect of a permutation directly. We prove that CBC mode is still secure if an encrypted counter is used in place of a random string as the IV for each message. Finally, we show that the 'ciphertext stealing' technique is secure.

For CFB, we first discuss the work of [AGPS01], who offer a proof for both CFB mode and an optimized variant which enhances the error-recovery properties of standard CFB. We show that their proof is incomplete, as shown in sections 4.2. We then offer our own proof. We show that full-width CFB is secure if the IV is any 'generalized counter', and that both full-width and truncated t-bit CFB are secure if the IV is an encrypted counter. We also show that, unlike CBC mode, it is safe to 'carry over' the final shift-register value from the previous message as the IV for the next message.

Finally, OFB mode is in fact a simple modification to CFB mode, and we prove the security of OFB by relating it to CFB.

Mode	Section	Notion	Security with	
			(t, q, ε)-**PRF**	(t, q, ε)-**PRP**
CBC	3	LOR-CPA	—	$2\varepsilon + \dfrac{q(q-1)}{2^\ell - q}$
CFB	4	LOR-CPA	$2\varepsilon + \dfrac{q(q-1)}{2^\ell}$	$2\varepsilon + \dfrac{q(q-1)}{2^{\ell-1}}$
OFB	5	LOR-CPA	$2\varepsilon + \dfrac{q(q-1)}{2^\ell}$	$2\varepsilon + \dfrac{q(q-1)}{2^{\ell-1}}$

Table 1.1: Summary of our results. In all cases, q is the number of block cipher applications used in the game.

As a convenient guide, our security bounds are summarized in table 1.1.

1.4 The rest of the paper

In section 2 we define the various bits of notation and terminology we'll need in the rest of the paper. The formal definitions are given for our new 'result-or-garbage' security notion, and for our generalized counters. In section 3 we study CBC mode, and ciphertext stealing. In section 4 we study CFB mode. In section 5 we study OFB mode.

2 Notation and definitions

2.1 Bit strings

Most of our notation for bit strings is standard. The main thing to note is that everything is zero-indexed.

- We write $\Sigma = \{0, 1\}$ for the set of binary digits. Then Σ^n is the set of n-bit strings, and Σ^* is the set of all (finite) bit strings.
- If x is a bit string then $|x|$ is the length of x. If $x \in \Sigma^n$ then $|x| = n$.
- If $x, y \in \Sigma^n$ are strings of bits of the same length then $x \oplus y \in \Sigma^n$ is their bitwise XOR.
- If x is a bit string and i is an integer satisfying $0 \leqslant i < |x|$ then $x[i]$ is the ith bit of x. If a and b are integers satisfying $0 \leqslant a \leqslant b \leqslant |x|$ then $x[a \mathinner{..} b]$ is the substring of x beginning with bit a and ending just *before* bit b. We have $|x[i]| = 1$ and $|x[a \mathinner{..} b]| = b - a$; if $y = x[a \mathinner{..} b]$ then $y[i] = x[a + i]$.
- If x and y are bit strings then xy is the result of concatenating y to x. If $z = xy$ then $|z| = |x| + |y|$; $z[i] = x[i]$ if $0 \leqslant i < |x|$ and $z[i] = y[i - |x|]$ if $|x| \leqslant i < |x| + |y|$. Sometimes, for clarity (e.g., to distinguish from integer multiplication) we write $x \parallel y$ instead of xy.
- The empty string is denoted λ. We have $|\lambda| = 0$, and $x = x\lambda = \lambda x$ for all strings $x \in \Sigma^*$.
- If x is a bit string and n is a natural number then x^n is the result of concatenating x to itself n times. We have $x^0 = \lambda$ and if $n > 0$ then $x^n = x^{n-1} \parallel x = x \parallel x^{n-1}$.

- If x and y are bit strings, $|x| = \ell$, and $|y| = t$, then we define $x \ll_t y$ as:

$$x \ll_t y = (xy)[t \mathinner{..} t+\ell] = \begin{cases} x[t \mathinner{..} \ell] \parallel y & \text{if } t < \ell \text{, or} \\ y[t-\ell \mathinner{..} t] & \text{if } t \geqslant \ell. \end{cases}$$

  Observe that, if $z = x \ll_t y$ then $|z| = |x| = \ell$ and

$$z[i] = (xy)[i+t] = \begin{cases} x[i+t] & \text{if } 0 \leqslant i < \ell - t \text{, or} \\ y[i+t-\ell] & \text{if } \min(0, \ell - t) \leqslant i < \ell. \end{cases}$$

  Obviously $x \ll_0 \lambda = x$, and if $|x| = |y| = t$ then $x \ll_t y = y$. Finally, if $|y| = t$ and $|z| = t'$ then $(x \ll_t y) \ll_{t'} z = x \ll_{t+t'} (yz)$.

2.2 Other notation

- The symbol $\perp$ ('bottom') is a value different from every bit string.
- We write $\mathcal{F}^{l,L}$ as the set of all functions from Σ^l to Σ^L, and $\mathcal{P}^l$ as the set of all permutations on Σ^l.

2.3 Algorithm descriptions

An *adversary* is a probabilistic algorithm which attempts (possibly) to 'break' a cryptographic scheme. We will often provide adversaries with oracles which compute values with secret data. The *running time* of an adversary conventionally includes the size of the adversary's description: this is an attempt to 'charge' the adversary for having large precomputed tables.

Most of the notation used in the algorithm descriptions should be obvious. We briefly note a few features which may be unfamiliar.

- The notation $a \leftarrow x$ denotes the action of assigning the value x to the variable a.
- We write oracles as superscripts, with dots marking where inputs to the oracle go, e.g., $A^{O(\cdot)}$.
- The notation $a \xleftarrow{R} X$, where X is a finite set, denotes the action of assigning to a a random value $x \in X$ according to the uniform probability distribution on X; i.e., following $a \xleftarrow{R} X$, we have $\Pr[a = x] = 1/|X|$ for any $x \in X$.

The notation is generally quite sloppy about types and scopes. We don't think these informalities cause much confusion, and they greatly simplify the presentation of the algorithms.

2.4 Pseudorandom functions and permutations

Our definitions of pseudorandom functions and permutations are standard. We provide them for the sake of completeness.

2.4.1 Definition (Pseudorandom function family) A *pseudorandom function family (PRF)* $F = \{F_K\}_K$ is a collection of functions $F_K\colon \Sigma^\ell \to \Sigma^L$ indexed by a *key* $K \in \operatorname{keys} F$. If A is any adversary, we define A's *advantage in distinguishing F from a random function* to be

$$\mathbf{Adv}_F^{\mathrm{prf}}(A) = \Pr[K \xleftarrow{R} \operatorname{keys} F : A^{F_K(\cdot)} = 1] - \Pr[R \xleftarrow{R} \mathcal{F}^{\ell,L} : A^{R(\cdot)} = 1]$$

where the probability is taken over all choices of keys, random functions, and the internal coin-tosses of A. The *insecurity of* F is given by

$$\mathbf{InSec}^{\mathrm{prf}}(F; t, q) = \max_A \mathbf{Adv}^{\mathrm{prf}}_F(A)$$

where the maximum is taken over all adversaries which run in time t and issue at most q oracle queries. If $\mathbf{InSec}^{\mathrm{prf}}(F; t, q) \leqslant \varepsilon$ then we say that F is a (t, q, ε)-PRF. $\square$

2.4.2 Definition (Pseudorandom permutation family) A *pseudorandom permutation family (PRP)* $E = \{E_K\}_K$ is a collection of permutations $E_K\colon \Sigma^\ell \to \Sigma^\ell$ indexed by a *key* $K \in \operatorname{keys} E$. If A is any adversary, we define A's *advantage in distinguishing* E *from a random permutation* to be

$$\mathbf{Adv}^{\mathrm{prp}}_F(A) = \Pr[K \xleftarrow{R} \operatorname{keys} E : A^{E_K(\cdot)} = 1] - \Pr[P \xleftarrow{R} \mathcal{P}^\ell : A^{P(\cdot)} = 1]$$

where the probability is taken over all choices of keys, random permutations, and the internal coin-tosses of A. Note that the adversary is not allowed to query the inverse permutation $E_K^{-1}(\cdot)$ or $P^{-1}(\cdot)$. The *insecurity of* E is given by

$$\mathbf{InSec}^{\mathrm{prp}}(E; t, q) = \max_A \mathbf{Adv}^{\mathrm{prf}}_E(A)$$

where the maximum is taken over all adversaries which run in time t and issue at most q oracle queries. If $\mathbf{InSec}^{\mathrm{prp}}(E; t, q) \leqslant \varepsilon$ then we say that E is a (t, q, ε)-PRP. $\square$

The following result is standard; we shall require it for the security proofs of CFB and OFB modes. The proof is given as an introduction to our general approach.

2.4.3 Proposition *Suppose E is a PRP over Σ^ℓ. Then*

$$\mathbf{InSec}^{\mathrm{prf}}(E; t, q) \leqslant \mathbf{InSec}^{\mathrm{prp}}(E; t, q) + \frac{q(q-1)}{2^{\ell+1}}.$$

Proof We claim

$$\mathbf{InSec}^{\mathrm{prf}}(\mathcal{P}^\ell; t, q) \leqslant \frac{q(q-1)}{2^{\ell+1}},$$

i.e., that a *perfect* ℓ-bit random permutation is a PRF with the stated bounds. The proposition follows immediately from this claim and the definition of a PRP.

We now prove the claim. Consider any adversary A. Let x_i be A's queries, and let y_i be the responses, for $0 \leqslant i < q$. Assume, without loss of generality, that the x_i are distinct. Let C_n be the event in the random-function game $\mathbf{Expt}^{\mathrm{prf}\text{-}0}_{\mathcal{P}^\ell}(A)$ that $y_i = y_j$ for some i and j where $0 \leqslant i < j < n$. Then

$$\Pr[C_n] \leqslant \sum_{0 \leqslant i < n} \frac{i}{2^\ell} = \frac{n(n-1)}{2^{\ell+1}}. \tag{1}$$

It's clear that the two games proceed identically if C_q doesn't occur in the random-function game. The claim follows. $\square$

2.5 Symmetric encryption

We begin with a purely syntactic description of a symmetric encryption scheme, and then define our two notions of security.

2.5.1 Definition (Symmetric encryption) A *symmetric encryption scheme* is a triple of algorithms $\mathcal{E} = (G, E, D)$, with three (implicitly) associated sets: a keyspace, a plaintext space, and a ciphertext space.

- G is a probabilistic *key-generation algorithm*. It is invoked with no arguments, and returns a key K which can be used with the other two algorithms. We write $K \leftarrow G()$.
- E is a probabilistic *encryption algorithm*. It is invoked with a key K and a *plaintext* x in the plaintext space, and it returns a *ciphertext* y in the ciphertext space. We write $y \leftarrow E_K(x)$.
- D is a deterministic *decryption algorithm*. It is invoked with a key K and a ciphertext y, and it returns either a plaintext x or the distinguished symbol $\perp$. We write $x \leftarrow D_K(y)$.

For correctness, we require that whenever y is a possible result of computing $E_K(x)$, then $x = D_K(y)$. □

Our primary notion of security is *left-or-right indistinguishability under chosen-plaintext attack* (LOR-CPA), since it offers the best reductions to the other common notions. (We can't achieve security against chosen ciphertext attack using any of our modes, so we don't even try.) See [BDJR97] for a complete discussion of LOR-CPA, and how it relates to other security notions for symmetric encryption.

2.5.2 Definition (Left-or-right indistinguishability) Let $\mathcal{E} = (G, E, D)$ be a symmetric encryption scheme. Define the function $lr(b, x_0, x_1) = x_b$. Then for any adversary A, we define A's *advantage against the LOR-CPA security of* $\mathcal{E}$ as

$$\mathbf{Adv}_{\mathcal{E}}^{\text{lor-cpa}}(A) = \Pr[K \leftarrow G() : A^{E_K(lr(1,\cdot,\cdot))} = 1] - \Pr[K \leftarrow G() : A^{E_K(lr(0,\cdot,\cdot))} = 1].$$

We define the *LOR-CPA insecurity of* $\mathcal{E}$ to be

$$\mathbf{InSec}^{\text{lor-cpa}}(\mathcal{E}; t, q_E, \mu_E) = \max_A \mathbf{Adv}_{\mathcal{E}}^{\text{lor-cpa}}(A)$$

where the maximum is taken over all adversaries which run in time t and issue at most q_E encryption queries totalling at most μ_E bits. If $\mathbf{InSec}^{\text{lor-cpa}}(\mathcal{E}; t, q_E, \mu_E) \leqslant \varepsilon$ then we say that $\mathcal{E}$ is $(t, q_E, \mu_E, \varepsilon)$-LOR-CPA. □

Our second notion is named *result-or-garbage* and abbreviated ROG-CPA. It is related to the notion used by [RBBK01], though different in important ways: for example, there are reductions both ways between ROG-CPA and LOR-CPA (and hence the other standard notions of security for symmetric encryption), whereas the notion of [RBBK01] is strictly stronger than LOR-CPA. Our idea is that an encryption scheme is secure if ciphertexts of given plaintexts – *results* – hard to distinguish from strings constructed independently of any plaintexts – *garbage*. We formalize this notion in terms of a *garbage-emission algorithm* W which is given only the length of the plaintext. The algorithm W will usually be probabilistic, and may maintain state. Unlike [RBBK01], we don't require that W's output 'look random' in any way, just that it be chosen independently of the adversary's plaintext selection.

2.5.3 Definition (Result-or-garbage indistinguishability) Let $\mathcal{E} = (G, E, D)$ be a symmetric encryption scheme, and let W be a possibly-stateful, possibly-probabilistic *garbage-emission* algorithm. Then for any adversary A, we define A's *advantage against the ROG-CPA-W security of* $\mathcal{E}$ as

$$\mathbf{Adv}_{\mathcal{E}}^{\text{rog-cpa-}W}(A) = \Pr[K \leftarrow G() : A^{E_K(\cdot)} = 1] - \Pr[A^{W(|\cdot|)} = 1].$$

We define the *ROG-CPA insecurity of $\mathcal{E}$* to be

$$\mathbf{InSec}^{\text{lor-cpa}}(\mathcal{E}; t, q_E, \mu_E) = \max_A \mathbf{Adv}_{\mathcal{E}}^{\text{lor-cpa}}(A)$$

where the maximum is taken over all adversaries which run in time t and issue at most q_E encryption queries totalling at most μ_E bits. If $\mathbf{InSec}^{\text{rog-cpa-}W}(\mathcal{E}; t, q_E, \mu_E) \leqslant \varepsilon$ for some W then we say that $\mathcal{E}$ is $(t, q_E, \mu_E, \varepsilon)$-ROG-CPA. □

The following proposition relates our new notion to the existing known notions of security.

2.5.4 Proposition (ROG $\Leftrightarrow$ LOR) *Let $\mathcal{E}$ be a symmetric encryption scheme. Then,*

1. *for all garbage-emission algorithms W,*

$$\mathbf{InSec}^{\text{lor-cpa}}(\mathcal{E}; t, q_E, \mu_E) \leqslant 2 \cdot \mathbf{InSec}^{\text{rog-cpa-}W}(\mathcal{E}; t + t_E\mu_E, q_E, \mu_E)$$

and

2. *there exists a garbage-emission algorithm W_{ROR} for which*

$$\mathbf{InSec}^{\text{rog-cpa-}W_{\text{ROR}}}(\mathcal{E}; t, q_E, \mu_E) \leqslant \mathbf{InSec}^{\text{lor-cpa}}(\mathcal{E}; t + t_E\mu_E, q_E, \mu_E)$$

for some fairly small constant t_E.

2.5.5 Remark Note the asymmetry between these two statements. ROG-CPA-W implies LOR-CPA for *any* garbage emitter W, but LOR-CPA implies ROG-CPA-W_{ROR} for the specific emitter W_{ROR} only. □

Proof of proposition 2.5.4 1. Let W and $\mathcal{E}$ be given, and let A be an adversary attacking the LOR-CPA security of $\mathcal{E}$. Consider adversary B attacking $\mathcal{E}$'s ROG-CPA-W security.

Adversary $B^E(\cdot)$:	Function $lr(b, x_0, x_1)$:
$b^* \xleftarrow{R} \Sigma$;	**if** $b = 0$ **then return** x_0 **else return** x_1;
$b \leftarrow A^{E(lr(b^*, \cdot, \cdot))}$;	
if $b = b^*$ **then return** 1 **else return** 0;	

If $E(\cdot)$ is the 'result' encryption oracle, then B simulates the left-or-right game for the benefit of A, and therefore returns 1 with probability $(\mathbf{Adv}_{\mathcal{E}}^{\text{lor-cpa}}(A) + 1)/2$. On the other hand, if $E(\cdot)$ returns 'garbage' then the oracle responses are entirely independent of b^*. This follows because A is constrained to query only on pairs of plaintexts with equal lengths, and the responses are dependent only on these (common) lengths and any internal state and coin tosses of W. So b is independent of b^* and $\Pr[b = b^*] = \frac{1}{2}$. The result follows.

2. Let $\mathcal{E} = (G, E, D)$ be given. Our garbage emitter simulates the real-or-random game of [BDJR97]. Let $K_W = \perp$ initially: we define our emitter W_{ROR} thus:

Garbage emitter $W_{\text{ROR}}(n)$:
  if $K_W = \perp$ **then** $K_W \leftarrow G()$;
  $x \xleftarrow{R} \Sigma^n$;
  return $E_{K_W}(x)$;

We now show that $\mathbf{InSec}^{\text{rog-cpa-}W_{\text{ROR}}}(\mathcal{E}; t, q_E, \mu_E) \leqslant \mathbf{InSec}^{\text{lor-cpa}}(\mathcal{E}; t + t_E\mu_E, q_E, \mu_E)$ for our W_{ROR}. Let A be an adversary attacking the ROG-CPA-W_{ROR} security of $\mathcal{E}$. Consider adversary B attacking $\mathcal{E}$'s LOR-CPA security:

Adversary $B^{E(\cdot,\cdot)}$:	Function *lorify*(x):
$b \leftarrow A^{\mathit{lorify}(\cdot)}$;	$x' \xleftarrow{R} \Sigma^{\lvert x \rvert}$;
return b;	**return** $E(x', x)$;

The adversary simulates the ROG-CPA-W_{ROR} games perfectly, since the game has chosen the random K_W for us already: the 'left' game returns only the results of encrypting random 'garbage' plaintexts x', while the right game returns correct results of encrypting the given plaintexts x. The result follows. □

2.6 Initialization vectors and encryption modes

In order to reduce the number of definitions in this paper to a tractable level, we will describe the basic modes independently of how initialization vectors (IVs) are chosen, and then construct the actual encryption schemes by applying various IV selection methods from the modes.

We consider the following IV selection methods.

Random selection An IV is chosen uniformly at random just before encrypting each message.

Counter The IV for each message is a *generalized counter* (see discussion below, and definition 2.6.1).

Encrypted counter The IV for a message is the result of applying the block cipher to a generalized counter, using the same key as for message encryption.

Carry-over The IV for the first message is fixed; the IV for subsequent messages is some function of the previous plaintexts or ciphertexts (e.g., the last ciphertext block of the previous message).

Not all of these methods are secure for all of the modes we consider.

2.6.1 Definition (Generalized counters) If S is a finite set, then a *generalized counter in S* is an bijection $c\colon \{0, 1, \ldots, |S| - 1\} \leftrightarrow S$. For brevity, we shall refer simply to 'counters', leaving implicit the generalization. □

2.6.2 Remark (Examples of generalized counters)

- There is a 'natural' binary representation of the natural numbers $\{0, 1, \ldots, 2^\ell - 1\}$ as ℓ-bit strings: for any $n \in \{0, 1, \ldots, 2^\ell - 1\}$, let $R(n)$ be the unique $r \in \Sigma^\ell$ such that $n = \sum_{0 \leqslant i < \ell} 2^i r[i]$. Then $R(\cdot)$ is a generalized counter in Σ^ℓ.

- We can represent elements of the finite field $\mathbb{F}_{2^\ell}$ as ℓ-bit strings. Let $p(x) \in \mathbb{F}_2[x]$ be a primitive polynomial of degree ℓ; then represent $\mathbb{F}_{2^\ell}$ by $\mathbb{F}_2[x]/(p(x))$. Now for any $a \in \mathbb{F}_{2^\ell}$, let $R(a)$ be the unique $r \in \Sigma^\ell$ such that $a = \sum_{0 \leqslant i < \ell} x^i r[i]$. Because $p(x)$ is primitive, x generates the multiplicative group $\mathbb{F}^*_{2^\ell}$, so define $c(n) = R(x^n)$ for $0 \leqslant n < 2^\ell - 1$ and $c(2^{\ell-1}) = 0^\ell$; then $c(\cdot)$ is a generalized counter in Σ^ℓ. This counter can be implemented efficiently in hardware using a linear feedback shift register. □

2.6.3 Definition (Encryption modes) A *block cipher encryption mode* $m_P = (\mathit{encrypt}, \mathit{decrypt})$ is a pair of deterministic oracle algorithms (and implicitly-defined plaintext and ciphertext spaces) which satisfy the following conditions:

1. The algorithm *encrypt* runs with oracle access to a permutation $P\colon \Sigma^\ell \leftrightarrow \Sigma^\ell$; on input a

plaintext x and an initialization vector $v \in \Sigma^\ell$, it returns a ciphertext y and a *chaining value* $v' \in \Sigma^\ell$. We write $(v', y) = \mathit{encrypt}^{P(\cdot)}(v, x)$.

2. The algorithm *decrypt* runs with oracle access to a permutation $P\colon \Sigma^\ell \leftrightarrow \Sigma^\ell$ and its inverse $P^{-1}(\cdot)$; on input a ciphertext y and an initialization vector $v \in \Sigma^\ell$, it returns a plaintext x. We write that $x = \mathit{decrypt}^{P(\cdot),P^{-1}(\cdot)}(v, y)$.

3. For all permutations $P\colon \Sigma^\ell \leftrightarrow \Sigma^\ell$, all plaintexts x and all initialization vectors v, if $(v', y) = \mathit{encrypt}^{P(\cdot)}(v, x)$ then $x = \mathit{decrypt}^{P(\cdot),P^{-1}(\cdot)}(v, y)$.

Similarly, a *PRF encryption mode* $m_F = (\mathit{encrypt}, \mathit{decrypt})$ is a pair of deterministic oracle algorithms (and implicitly-defined plaintext and ciphertext spaces) which satisfy the following conditions:

1. The algorithm *encrypt* runs with oracle access to a function $F\colon \Sigma^\ell \to \Sigma^L$; on input a plaintext x and an initialization vector $v \in \Sigma^\ell$, it returns a ciphertext y and a *chaining value* $v' \in \Sigma^\ell$. We write $(v', y) = \mathit{encrypt}^{F(\cdot)}(v, x)$.

2. The algorithm *decrypt* runs with oracle access to a function $F\colon \Sigma^\ell \to \Sigma^L$; on input a ciphertext y and an initialization vector $v \in \Sigma^\ell$, it returns a plaintext x. We write that $(v', x) = \mathit{decrypt}^{F(\cdot)}(v, y)$.

3. For all functions $F\colon \Sigma^\ell \to \Sigma^L$, all plaintexts x and all initialization vectors v, if $(v', y) = \mathit{encrypt}^{F(\cdot)}(v, x)$ then $x = \mathit{decrypt}^{F(\cdot)}(v, y)$. $\square$

2.6.4 Definition (Symmetric encryption schemes from modes) Let F be a pseudorandom permutation on Σ^ℓ (resp. a pseudorandom function from Σ^ℓ to Σ^L); let $m = (\mathit{encrypt}, \mathit{decrypt})$ be a block cipher (resp. PRF) encryption mode. (To save on repetition, if F is a PRF then define $F_K^{-1}(x) = \bot$ for all keys K and inputs x.) We define the following symmetric encryption schemes according to how IVs are selected.

- Randomized selection: define $\mathcal{E}\text{-}m\$^F = (G\text{-}m\$^F, E\text{-}m\$^F, D\text{-}m\$^F)$, where

Algorithm $G\text{-}m\$^F()$: $K \xleftarrow{R} \mathrm{keys}\, F$; **return** K;	Algorithm $E\text{-}m\$^F(K, x)$: $v \xleftarrow{R} \Sigma^\ell$; $(v', x) \leftarrow \mathit{encrypt}^{F_K(\cdot)}(v, x)$; **return** (v, y);	Algorithm $D\text{-}m\$^F(K, y')$; $(v, y) \leftarrow y'$; $(v', x) \leftarrow$ $\mathit{decrypt}^{F_K(\cdot),F_K^{-1}(\cdot)}(v, y)$; **return** x;

- Generalized counters: define $\mathcal{E}\text{-}mC^{F,c} = (G\text{-}mC^{F,c}, E\text{-}mC^{F,c}, D\text{-}mC^{F,c})$, where c is a generalized counter in Σ^ℓ, and

Algorithm $G\text{-}mC^{F,c}()$: $K \xleftarrow{R} \mathrm{keys}\, F$; $\mathit{i\text{-}msg} \leftarrow 0$; **return** K;	Algorithm $E\text{-}mC^{F,c}(K, x)$: $i \leftarrow c(\mathit{i\text{-}msg})$; $(v', x) \leftarrow \mathit{encrypt}^{F_K(\cdot)}(i, x)$; $\mathit{i\text{-}msg} \leftarrow \mathit{i\text{-}msg} + 1$; **return** (i, y);	Algorithm $D\text{-}mC^{F,c}(K, y')$; $(i, y) \leftarrow y'$; $(v', x) \leftarrow$ $\mathit{decrypt}^{F_K(\cdot),F_K^{-1}(\cdot)}(i, y)$; **return** x;

- Encrypted counters: if $L \geqslant \ell$, then define $\mathcal{E}\text{-}mE^{F,c} = (G\text{-}mE^{F,c}, E\text{-}mE^{F,c}, D\text{-}mE^{F,c})$, where c is a generalized counter in Σ^ℓ, and

Algorithm $G\text{-}mE^{F,c}()$:
$K \xleftarrow{R} \text{keys } F$;
$i\text{-}msg \leftarrow 0$;
return K;

Algorithm $E\text{-}mE^{F,c}(K, x)$:
$i \leftarrow c(i\text{-}msg)$;
$v \leftarrow F_K(i)[0 \mathinner{..} \ell]$;
$(v', x) \leftarrow \mathit{encrypt}^{F_K(\cdot)}(v, x)$;
$i\text{-}msg \leftarrow i\text{-}msg + 1$;
return (i, y);

Algorithm $D\text{-}mE^{F,c}(K, y')$;
$(i, y) \leftarrow y'$;
$v \leftarrow F_K(i)[0 \mathinner{..} \ell]$;
$(v', x) \leftarrow$
$\mathit{decrypt}^{F_K(\cdot), F_K^{-1}(\cdot)}(v, y)$;
return x;

(We require $L \geqslant \ell$ for this to be well-defined; otherwise the encrypted counter value is too short.)

- Carry-over: define $\mathcal{E}\text{-}mL^{F,V_0} = (G\text{-}mL^{F,V_0}, E\text{-}mL^{F,V_0}, D\text{-}mL^{F,V_0})$ where $V_0 \in \Sigma^\ell$ is the initialization vector for the first message, and

Algorithm $G\text{-}mL^{F,V_0}()$:
$K \xleftarrow{R} \text{keys } F$;
$v\text{-}next \leftarrow V_0$;
return K;

Algorithm $E\text{-}mL^{F,V_0}(K, x)$:
$v \leftarrow v\text{-}next$;
$(v', x) \leftarrow \mathit{encrypt}^{F_K(\cdot)}(v, x)$;
$v\text{-}next \leftarrow v'$;
return (v, y);

Algorithm $D\text{-}mL^{F,V_0}(K, y')$;
$(v, y) \leftarrow y'$;
$(v', x) \leftarrow$
$\mathit{decrypt}^{F_K(\cdot), F_K^{-1}(\cdot)}(v, y)$;
return x;

Note that, while the encryption algorithms of the above schemes are either randomized or stateful, the decryption algorithms are simple and deterministic. □

The following simple and standard result will be very useful in our proofs.

2.6.5 Proposition

1. *Suppose that $\mathcal{E}^P = (G^P, E^P, D^P)$ is one of the symmetric encryption schemes of definition 2.6.4, constructed from a pseudorandom permutation $P\colon \Sigma^\ell \leftrightarrow \Sigma^\ell$. If q is an upper bound on the number of PRP applications required for the encryption q_E messages totalling μ_E bits, then there is some small constant t' such that*

$$\mathbf{InSec}^{\text{lor-cpa}}(\mathcal{E}^P; t, q_E, \mu_E) \leqslant \mathbf{InSec}^{\text{lor-cpa}}(\mathcal{E}^{\mathcal{P}^\ell}; t, q_E, \mu_E) + 2 \cdot \mathbf{InSec}^{\text{prp}}(P; t + qt', q).$$

2. *Similarly, suppose that $\mathcal{E}^F = (G^F, E^F, D^F)$ is one of the symmetric encryption schemes of definition 2.6.4, constructed from a pseudorandom function $F\colon \Sigma^\ell \to \Sigma^L$. If q is an upper bound on the number of PRP applications required for the encryption q_E messages totalling μ_E bits, then there is some small constant t' such that*

$$\mathbf{InSec}^{\text{lor-cpa}}(\mathcal{E}^F; t, q_E, \mu_E) \leqslant \mathbf{InSec}^{\text{lor-cpa}}(\mathcal{E}^{\mathcal{F}^{\ell,L}}; t, q_E, \mu_E) + 2 \cdot \mathbf{InSec}^{\text{prf}}(F; t + qt', q).$$

Proof 1. Let A be an adversary attacking the LOR-CPA security of $\mathcal{E}^P$, which takes time t and issues q_E encryption queries totalling μ_E bits. We construct an adversary B attacking the security of the PRP P as follows. B selects a random $b^* \in_R \Sigma$. It then runs A, simulating the LOR-CPA game by using b^* to decide whether to encrypt the left or right plaintext, and using its oracle access to P to do the encryption. Eventually, A returns a bit b. If $b = b^*$, B returns 1 (indicating 'pseudorandom'); otherwise it returns 0.

If B's oracle is selected from the PRP P, then B correctly simulates the LOR-CPA game for $\mathcal{E}^P$, and B returns 1 with probability precisely $(\mathbf{Adv}^{\text{lor-cpa}}_{\mathcal{E}^P}(A) + 1)/2$. Conversely, if B's oracle is a random permutation, then B correctly simulates the LOR-CPA game for $\mathcal{E}^{\mathcal{P}^\ell}$, so B returns 1 with probability $(\mathbf{Adv}^{\text{lor-cpa}}_{\mathcal{E}^P}(A) + 1)/2$. Thus, we have

$$\mathbf{Adv}^{\text{prp}}_P(B) = (\mathbf{Adv}^{\text{lor-cpa}}_{\mathcal{E}^P}(A) + 1)/2 - (\mathbf{Adv}^{\text{lor-cpa}}_{\mathcal{E}^{\mathcal{P}^\ell}}(A) + 1)/2 \tag{2}$$

$$= (\mathbf{Adv}^{\text{lor-cpa}}_{\mathcal{E}^P}(A) - \mathbf{Adv}^{\text{lor-cpa}}_{\mathcal{E}^{\mathcal{P}^\ell}}(A))/2. \tag{3}$$

Note that the extra work which B does over A – initialization, tidying up and encrypting messages – is bounded by some small constant t_P multiplied by the number of oracle queries q made by B, and the required result follows by multiplying through by 2 and rearranging.

2. The proof for this case is almost identical: merely substitute F for P, 'PRF' for 'PRP' and $\mathcal{F}^{\ell,L}$ for $\mathcal{P}^\ell$ in the above. □

Of course, proving theorems about each of the above schemes individually will be very tedious. We therefore define a 'hybrid' scheme which switches between the above selection methods. This isn't a practical encryption scheme – just a 'trick' to reduce the number of complicated proofs we need to give.

2.6.6 Definition (Hybrid encryption modes) Let n_L, n_C and n_E be nonnegative integers, with $n_L + n_C + n_E \leqslant 2^\ell$; let F be a pseudorandom permutation on Σ^ℓ (resp. a pseudorandom function from Σ^ℓ to Σ^L); let $m = (\mathit{encrypt}, \mathit{decrypt})$ be a block cipher (resp. PRF) encryption mode let $V_0 \in \Sigma^\ell$ be an initialization vector; and let $c\colon \{0, 1, \ldots, 2^\ell - 1\} \to \Sigma^\ell$ be a generalized counter. (Again, if F is a PRF, we set $F_K(x) = \perp$ for all K and x.) We define the scheme $\mathcal{E}\text{-}mH^{F,V_0,c}_{n_L,n_C,n_E} = (G\text{-}mH^{F,V_0,c}_{n_L,n_C,n_E}, E\text{-}mH^{F,V_0,c}_{n_L,n_C,n_E}, D\text{-}mH^{F,V_0,c}_{n_L,n_C,n_E})$ as follows.

Algorithm $G\text{-}mH^{F,V_0,c}_{n_L,n_C,n_E}()$:
 $K \xleftarrow{R} \text{keys}\,F$;
 $\mathit{i\text{-}msg} \leftarrow 0$;
 $\mathit{v\text{-}next} \leftarrow V_0$;
 return K;

Algorithm $D\text{-}mH^{F,V_0,c}_{n_L,n_C,n_E}(K, y')$;
 $(v, y) \leftarrow y'$;
 $(v', x) \leftarrow \mathit{decrypt}^{F_K(\cdot), F_K^{-1}(\cdot)}(v, y)$;
 return x;

Algorithm $E\text{-}mH^{F,V_0,c}_{n_L,n_C,n_E}(K, x)$:
 if $\mathit{i\text{-}msg} < n_L$ **then** $v \leftarrow \mathit{v\text{-}next}$;
 else if $\mathit{i\text{-}msg} < n_L + n_C$ **then** $v \leftarrow c(\mathit{i\text{-}msg})$;
 else if $\mathit{i\text{-}msg} < n_L + n_C + n_E$ **then** $v \leftarrow F_K(c(\mathit{i\text{-}msg})[0 \mathinner{..} \ell])$;
 else $v \xleftarrow{R} \Sigma^\ell$;
 $(v', x) \leftarrow \mathit{encrypt}^{F_K(\cdot)}(v, x)$;
 $\mathit{v\text{-}next} \leftarrow v'$;
 $\mathit{i\text{-}msg} \leftarrow \mathit{i\text{-}msg} + 1$;
 return (v, y);

For this to be well-defined, we require that $L \geqslant \ell$ or $n_E = 0$ – otherwise the encrypted counter values are too short. □

The following proposition relates the security of our artificial hybrid scheme to that of the practical schemes defined in definition 2.6.4.

2.6.7 Proposition *Let F be a pseudorandom permutation on Σ^ℓ (resp. a pseudorandom function from Σ^ℓ to Σ^L); let m be a block cipher (resp. PRF) encryption mode. Then:*

1. $\mathbf{InSec}^{\text{lor-cpa}}(E\text{-}m\$^F; t, q_E, \mu_E) \leqslant \mathbf{InSec}^{\text{lor-cpa}}(E\text{-}mH^{F,V_0,c}_{0,0,0}; t, q_E, \mu_E)$
2. $\mathbf{InSec}^{\text{lor-cpa}}(E\text{-}mC^{F,c}; t, q_E, \mu_E) \leqslant \mathbf{InSec}^{\text{lor-cpa}}(E\text{-}mH^{F,V_0,c}_{q_E,0,0}; t, q_E, \mu_E)$
3. $\mathbf{InSec}^{\text{lor-cpa}}(E\text{-}mE^{F,c}; t, q_E, \mu_E) \leqslant \mathbf{InSec}^{\text{lor-cpa}}(E\text{-}mH^{F,V_0,c}_{0,q_E,0}; t, q_E, \mu_E)$

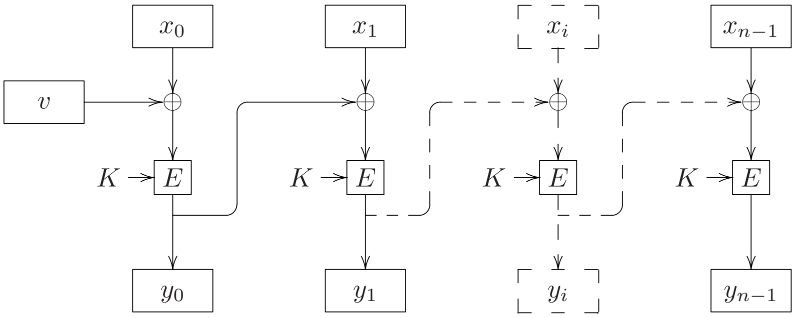

Figure 3.1: Encryption using CBC mode

4. $\mathbf{InSec}^{\text{lor-cpa}}(E\text{-}mL^{F,V_0}; t, q_E, \mu_E) \leqslant \mathbf{InSec}^{\text{lor-cpa}}(E\text{-}mH^{F,V_0,c}_{0,0,q_E}; t, q_E, \mu_E)$

Proof For 1, it suffices to observe that $E\text{-}m\$^F \equiv E\text{-}mH^{F,V_0,c}_{0,0,0}$ for any c, V_0. For 2, note that $E\text{-}mC^{F,c}$ behaves identically to $E\text{-}mH^{F,V_0,c}_{q_E,0,0}$ for any c, V_0 for the first q_E encryption queries; but no adversary is permitted to exceed this limit, and hence no adversary can distinguish the two. Similarly, for 4, note that $E\text{-}mL^{F,V_0}$ behaves identically to $E\text{-}mH^{F,V_0,c}_{0,0,q_E}$ for any c, V_0 for the first q_E encryption queries.

The case of 3 is slightly more complicated: $E\text{-}mE^{F,c}$ behaves identically to $E\text{-}mH^{F,V_0,c}_{0,q_E,0}$ for the first q_E encryption queries *except* that the latter returns different initialization vectors from its encryption oracle. However, since the counter c is fixed public knowledge, it is trivial to construct a fully faithful replica of the mE game given the hybrid oracle, such that no adversary can distinguish the two. □

3 Ciphertext block chaining (CBC) encryption

3.1 Description

Suppose E is an ℓ-bit pseudorandom permutation. CBC mode works as follows. Given a message X, we divide it into ℓ-bit blocks x_0, x_1, …, x_{n-1}. Choose an initialization vector $v \in \Sigma^\ell$. Before passing each x_i through E, we XOR it with the previous ciphertext, with v standing in for the first block:

$$y_0 = E_K(x_0 \oplus v) \qquad y_i = E_K(x_i \oplus y_{i-1} \text{ (for } 1 \leqslant i < n). \tag{4}$$

The ciphertext is then the concatenation of v and the y_i. Decryption is simple:

$$x_0 = E_K^{-1}(y_0) \oplus v \qquad x_i = E_K^{-1}(y_i) \oplus y_{i-1} \text{ (for } 1 \leqslant i < n) \tag{5}$$

See figure 3.1 for a diagram of CBC encryption.

3.1.1 Definition (CBC algorithms) For any permutation $P\colon \Sigma^\ell \to \Sigma^\ell$, any initialization vector $v \in \Sigma^\ell$, any plaintext $x \in \Sigma^{\ell\mathbb{N}}$ and any ciphertext $y \in \Sigma^*$, we define the encryption mode $CBC = (\mathit{cbc\text{-}encrypt}, \mathit{cbc\text{-}decrypt})$ as follows:

Algorithm *cbc-encrypt*$^{P(\cdot)}(v, x)$:
  $y \leftarrow \lambda$;
  for $i = 0$ **to** $|x|/\ell$ **do**
    $x_i \leftarrow x[\ell i \mathinner{..} \ell(i+1)]$;
    $y_i \leftarrow P(x_i \oplus v)$;
    $v \leftarrow y_i$;
    $y \leftarrow y \parallel y_i$;
  return (v, y);

Algorithm *cbc-decrypt*$^{P(\cdot), P^{-1}(\cdot)}(v, y)$:
  if $|y| \bmod \ell \neq 0$ **then return** $\perp$;
  $x \leftarrow \lambda$;
  for $1 = 0$ **to** $|y|/\ell$ **do**
    $y_i \leftarrow y[\ell i \mathinner{..} \ell(i+1)]$;
    $x_i \leftarrow P^{-1}(y_i) \oplus v$;
    $v \leftarrow y_i$;
    $x \leftarrow x \parallel x_i$;
  return (v, x);

Now, let c be a generalized counter in Σ^ℓ. We define the encryption schemes $E\text{-}CBC\$^P$, $E\text{-}CBCE^{P,c}$ and $E\text{-}CBCH^{P,c,\perp}_{0,0,n_E}$, as described in definition 2.6.4. □

3.2 Security of CBC mode

We now present our main theorem on CBC mode.

3.2.1 Theorem (Security of hybrid CBC mode) *Let $P\colon \mathrm{keys}_P \times \Sigma^\ell \to \Sigma^\ell$ be a pseudorandom permutation; let $V_0 \in \Sigma^\ell$ be an initialization vector; let $n_L \in \{0,1\}$; let c be a generalized counter in Σ^ℓ; and let $n_C \in \mathbb{N}$ be a nonnegative integer; and suppose that at most one of n_L and n_C is nonzero. Then, for any t, $q_E \geqslant n$ and μ_E,*

$$\mathbf{InSec}^{\text{lor-cpa}}(\mathcal{E}\text{-}CBCH^{P,c,V_0}_{n_L,0,n_E}; t, q_E, \mu_E) \leqslant 2 \cdot \mathbf{InSec}^{\text{prp}}(P; t + q t_P, q) + \frac{q(q-1)}{2^\ell - q}$$

where $q = n_L + n_E + \mu_E/\ell$ and t_P is some small constant.

The proof of this theorem we postpone until section 3.4. As promised, the security of our randomized and stateful schemes follow as simple corollaries.

3.2.2 Corollary (Security of practical CBC modes) *Let P and c be as in theorem 3.2.1. Then for any t, q_E and μ_E, and some small constant t_P,*

$$\mathbf{InSec}^{\text{lor-cpa}}(\mathcal{E}\text{-}CBC\$^P; t, q_E, \mu_E) \leqslant 2 \cdot \mathbf{InSec}^{\text{prp}}(P; t + q t_P, q) + \frac{q(q-1)}{2^\ell - q}$$
$$\mathbf{InSec}^{\text{lor-cpa}}(\mathcal{E}\text{-}CBCE^{P,c}; t, q_E, \mu_E) \leqslant 2 \cdot \mathbf{InSec}^{\text{prp}}(P; t + q' t_P, q') + \frac{q'(q'-1)}{2^\ell - q'}$$

and

$$\mathbf{InSec}^{\text{lor-cpa}}(\mathcal{E}\text{-}CBCL^{P,V_0}; t, 1, \mu_E) \leqslant 2 \cdot \mathbf{InSec}^{\text{prp}}(P; t + q t_P, q) + \frac{q(q-1)}{2^\ell - q}$$

where $q = \mu_E/\ell$, and $q' = q + q_E$.

Proof Follows from theorem 3.2.1 and proposition 2.6.7. □

3.2.3 Remark The insecurity of CBC mode over that inherent in the underlying PRP is essentially a birthday bound: note for $q \leqslant 2^{\ell/2}$, our denominator $2^\ell - q \approx 2^\ell$, and for larger q, the term $q(q-1)/2^\ell > 1$ anyway, so all security is lost (according to the above result). Compared to [BDJR97, theorem 17], we gain the tiny extra term in the denominator, but lose the PRP-as-a-PRF term $q^2/2^{\ell-1}$.[1] □

[1] In fact, they don't prove the stated bound of $q(3q-2)/2^{\ell+1}$ but instead the larger $q(2q-1)/2^\ell$. The error is in the application of their proposition 8: the PRF-insecurity term is doubled, so the PRP-as-a-PRF term must be also.

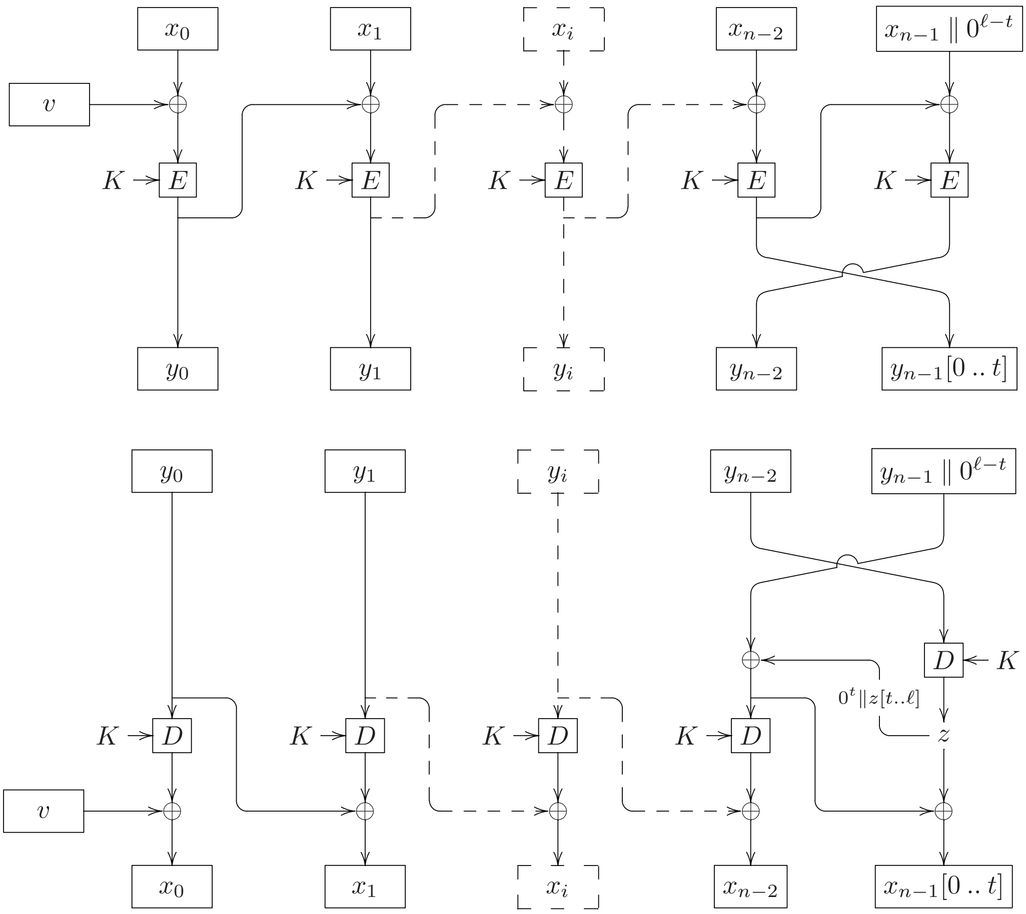

Figure 3.2: Encryption and decryption using CBC mode with ciphertext stealing

3.3 Ciphertext stealing

Ciphertext stealing [Dae95, Sch96, BR96] allows us to encrypt any message in Σ^* without the need for padding. The trick is to fill in the 'gap' at the end of the last block with the end bit of the previous ciphertext, and then to put the remaining short penultimate block at the end. Decryption proceeds by first decrypting the final block to recover the remainder of the penultimate one. See figure 3.2.

Encrypting messages shorter than the block involves 'IV stealing' – using the IV instead of the ciphertext from the last full-length block – which is a grotty hack but works fine if IVs are random; if the IVs are encrypted counters then there's nothing (modifiable) to steal from.

We formally present a description of a randomized CBC stealing mode.

3.3.1 Definition (CBC mode with ciphertext stealing) Let $P\colon \mathrm{keys}\, P \times \Sigma^\ell \to \Sigma^\ell$ be a pseudorandom permutation. Let c be a generalized counter on Σ^ℓ. We define the randomized symmetric encryption scheme $\mathcal{E}\text{-}\mathit{CBC\$\text{-}steal}^P = (\mathit{G\text{-}CBC\$\text{-}steal}^P, \mathit{E\text{-}CBC\$\text{-}steal}^P, \mathit{D\text{-}CBC\$\text{-}steal}^P)$ for messages in Σ^* as follows:

Algorithm $G\text{-}CBC\$\text{-}steal^P()$:
  $K \xleftarrow{R} \text{keys } P$;
  return K;

Algorithm $E\text{-}CBC\$\text{-}steal^P(K, x)$:
  $t \leftarrow |x| \bmod \ell$;
  if $t \neq 0$ **then** $x \leftarrow x \parallel 0^{\ell - t}$;
  $v \xleftarrow{R} \Sigma^\ell$;
  $y \leftarrow v \parallel \textit{cbc-encrypt}(K, v, x)$;
  if $t \neq 0$ **then**
    $b \leftarrow |y| - 2\ell$;
    $y \leftarrow y[0 \mathinner{..} b] \parallel y[b + \ell \mathinner{..} |y|] \parallel$
      $y[b \mathinner{..} b + t]$;
  return y;

Algorithm $D\text{-}CBC\$\text{-}steal^P(K, y)$:
  if $|y| < \ell$ **then return** $\perp$;
  $v \leftarrow y[0 \mathinner{..} \ell]$;
  $t = |y| \bmod \ell$;
  if $t \neq 0$ **then**
    $b \leftarrow |y| - t - \ell$;
    $z \leftarrow P_K^{-1}(y[b \mathinner{..} b + \ell])$;
    $y \leftarrow y[0 \mathinner{..} b] \parallel y[b + \ell \mathinner{..} |y|] \parallel$
      $z[t \mathinner{..} \ell]$;
  $x \leftarrow \textit{cbc-decrypt}(K, v, y[\ell \mathinner{..} |y|])$;
  if $t \neq 0$ **then**
    $x \leftarrow x \parallel z[0 \mathinner{..} t] \oplus y[b \mathinner{..} b + t]$;
  return x;

□

The security of ciphertext stealing follows directly from the definition and the security CBC mode.

3.3.2 Corollary (Security of CBC with ciphertext stealing) *Let* $P\colon \text{keys } P \times \Sigma^\ell \to \Sigma^\ell$ *be a pseudorandom permutation. Then*

$$\begin{aligned}\mathbf{InSec}^{\text{lor-cpa}}(\mathcal{E}\text{-}CBC\$\text{-}steal; t, q_E, \mu_E) &\leqslant \mathbf{InSec}^{\text{lor-cpa}}(\mathcal{E}\text{-}CBC\$; t, q_E, \mu_E + q_E(\ell - 1)) \\ &\leqslant 2 \cdot \mathbf{InSec}^{\text{prp}}(P; t + q t_P, q) + \frac{q(q-1)}{2^\ell - 2^{\ell/2}}\end{aligned}$$

where $q = \lfloor (\mu_E + q_E(\ell - 1))/\ell \rfloor$ *and* t_P *is some small constant.*

Proof From the definition, we see that the encryption algorithm *E-CBC-steal* simply pads a plaintext, encrypts it as for standard CBC mode, and postprocesses the ciphertext. Hence, if A is any adversary attacking $\mathcal{E}$*-CBC-steal*, we can construct an adversary A' which simply runs A except that, on each query to the encryption oracle, it pads both plaintexts, queries its CBC oracle, postprocesses the ciphertext returned, and gives the result back to A. The fact that plaintexts can now be up to $\ell - 1$ bits shorter than the next largest whole number of blocks means that B submits no more than $\mu_E + q_E(\ell - 1)$ bits of plaintext to its oracle. The required result follows then directly from theorem 3.2.1. □

3.4 Proof of theorem 3.2.1

The techniques and notation used in this proof will also be found in several of the others. We recommend that readers try to follow this one carefully.

We begin considering CBC mode using a completely random permutation. To simplify notation slightly, we shall write $n = n_L + n_E$. Our main goal is to prove the claim that there exists a garbage-emitter W such that

$$\mathbf{InSec}^{\text{rog-cpa-}W}(\mathcal{E}\text{-}CBCH^{\mathcal{P}^\ell, c, V_0}_{n_L, 0, n_E}; t, q_E, \mu_E) \leqslant \frac{q(q-1)}{2 \cdot (2^\ell - n)}.$$

From this, we can apply proposition 2.5.4 to obtain

$$\mathbf{InSec}^{\text{lor-cpa}}(\mathcal{E}\text{-}CBCH^{\mathcal{P}^\ell, c, \perp}_{0,0,n}; t, q_E, \mu_E) \leqslant \frac{q(q-1)}{2^\ell - n}.$$

Initialization:
  $i \leftarrow 0$;
  gone $\leftarrow \varnothing$;

Function *fresh*()
  $x \xleftarrow{R} \Sigma^\ell \setminus$ *gone*;
  gone $\leftarrow$ *gone* $\cup \{x\}$;
  return x;

Garbage emitter $W(m)$:
  if $i \geqslant 2^\ell$ **then abort**;
  if $i < n_L$ **then** $v \leftarrow V_0$;
  else if $i < n$ **then** $v \leftarrow$ *fresh*();
  $i \leftarrow i + 1$
  else $v \xleftarrow{R} \Sigma^\ell$;
  $y \leftarrow \lambda$;
  for $j = 0$ **to** m/ℓ;
    $y_j \leftarrow$ *fresh*();
    $y \leftarrow y \parallel y_j$;
  return (v, y);

Figure 3.3: Garbage emitter W for CBC mode

and, noting that there are precisely $q = \mu_E/\ell$ PRP-applications, we apply proposition 2.6.5 to obtain the required result.

Our garbage-emitter W is a bit complicated. It chooses random but *distinct* blocks for the 'ciphertext'; for the IVs, it uses V_0 for the first message if $n_L = 1$, and otherwise it chooses random blocks distinct from each other and the 'ciphertext' blocks for the next n_E messages, and just random blocks for subsequent ones. The algorithm W is shown in figure 3.3.

Fortunately, it doesn't need to be efficient: the above simulations only need to be able to do the LOR game, not the ROG one. The unpleasant-sounding **abort** only occurs after 2^ℓ queries. If that happens we give up and say the adversary won anyway: the claim is trivially true by this point, since the adversary's maximum advantage is 1.

Now we show that this lash-up is a good imitation of CBC encryption to someone who doesn't know the key. The intuition works like this: every time we query a random permutation at a new, fresh input value, we get a new, different, random output value; conversely, if we repeat an input, we get the same value out as last time. So, in the real 'result' CBC game, if all the permutation inputs are distinct, it looks just like the garbage emitted by W. Unfortunately, that's not quite enough: the adversary can work out what the permutation inputs ought to be and spot when there ought to have been a collision but wasn't. So we'll show that, provided all the P-inputs – values which *would* be input to the permutation if we're playing that game – are distinct, the two games look identical.

We need some notation to describe the values in the game. Let $c_i = c(i)$ be the ith counter value, for $0 \leqslant i < n_E$, and let v_i be the ith initialization vector, where $v_0 = V_0$ is as given if $n_L = 1$, $v_i = P(c_i - n_L)$ if $n_L \leqslant i < n$, and $v_i \in_R \Sigma^\ell$ if $n \leqslant i < q_E$. Let $q' = \mu_E/\ell = q - n$ be the total number of plaintext blocks in the adversary's queries, let x_i be the ith plaintext block queried, let y_i be the ith ciphertext block returned, let

$$w_i = \begin{cases} v_j & \text{if block } i \text{ is the first block of the } j\text{th query, and} \\ y_{i-1} & \text{otherwise} \end{cases}$$

and let $z_i = x_i \oplus w_i$, all for $0 \leqslant i < q'$. This is summarized diagrammatically in figure 3.4. The P-inputs are now precisely the c_i and the z_i. We'll denote probabilities in the 'result' game as $\Pr_R[\cdot]$ and in the 'garbage' game as $\Pr_G[\cdot]$.

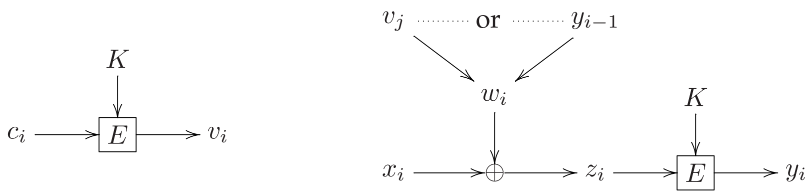

Figure 3.4: Notation for the proof of theorem 3.2.1.

Let C_r be the event, in either game, that $z_i = z_j$ for some $0 \leqslant i < j < r$, or that $z_i = c_j$ for some $0 \leqslant i < r$ and some $0 \leqslant j < n_E$. We need to bound the probability that $C_{q'}$ occurs in both the 'result' and 'garbage' games. We'll do this inductively. By the definition, $\Pr_R[C_0] = \Pr_G[C_0] = 0$.

Firstly, tweak the games so that all of the IVs corresponding to counters are chosen at the beginning, instead of as we go along. Obviously this doesn't make any difference to the adversary's view of the proceedings, but it makes our analysis easier.

Let's assume that C_r didn't happen; we want the probability that C_{r+1} did, which is obviously just the probability that z_r collides with some z_i for $0 \leqslant i < r$ or some c_i for $0 \leqslant i < n$. At this point, the previous z_i are fixed. So:

$$\Pr[C_{r+1}|\bar{C}_r] = \sum_{z \in \Sigma^\ell} \left(\sum_{0 \leqslant i < n} \Pr[z = c_i] + \sum_{0 \leqslant i < r} \Pr[z = z_i] \right) \cdot \Pr[z_r = z] \tag{6}$$

Now note that $z_r = w_r \oplus x_r$. We've no idea how x_r was chosen; but, one of the following cases holds.

1. If x_r is the first block of the first plaintext, i.e., $r = 0$, and $n_L = 1$, then $w_r = v_0$. However, in this case we know that $n_E = 0$ by hypothesis. There are no z_i which z_r might collide with, so the probability of a collision is zero.

2. If x_r is the first block of plaintext i, and $0 \leqslant i < n$, then $w_r = v_i$, and was chosen at random from a set of $2^\ell - i \leqslant 2^\ell - n \leqslant 2^\ell - n - r$ possibilities, either by our random permutation or by W. We know x_r is independent of w_r because none of the previous P-inputs were equal to c_i, by our assumption of $\bar{C}_r$.

3. If x_r is the first block of plaintext i, and $n \leqslant i < q'$, then $w_r = v_i$, and was chosen at random from all 2ℓ possible ℓ-bit blocks. We know x_r is independent of w_r because we just chose w_r at random, after x_r was chosen.

4. Otherwise, x_r is a subsequent block in some message, and $w_r = y_{r-1}$, and was chosen at random from a set of $2^\ell - n - r$ possibilities, either by our random permutation or by W. We know x_r is independent of w_r because z_{r-1} is a new P-input, by our assumption of $\bar{C}_r$.

So, except in case 1, which isn't a problem anyway, w_r is independent of x_r, and chosen uniformly at random from a set of at least $2^\ell - r - n$ elements, in either game – so we can already see that $\Pr_R[C_i] = \Pr_G[C_i]$ for any $i \geqslant 0$. Finally, the z_i and c_i are all distinct, so the

$z_i \oplus x$ and $c_i \oplus x$ must all be distinct, for any fixed x. So:

$$\Pr[C_{r+1}|\bar{C}_r] = \sum_{x\in\Sigma^\ell}\left(\sum_{0\leqslant i<n}\Pr[w_r = x\oplus c_i] + \sum_{0\leqslant i<r}\Pr[w_r = x\oplus z_i]\right)\cdot\Pr[x_r = x] \tag{7}$$

$$\leqslant \sum_{x\in\Sigma^\ell}\frac{r+n}{2^\ell - r - n}\Pr[x_r = x] = \frac{r+n}{2^\ell - r - n}\sum_{x\in\Sigma^\ell}\Pr[x_r = x] \tag{8}$$

$$= \frac{r+n}{2^\ell - r - n}. \tag{9}$$

Now we're almost home. All the c_i and z_i are distinct; all the v_i and y_i are random, assuming $C_{q'}$. We can bound $\Pr[C_{q'}]$ thus:

$$\Pr[C_{q'}] \leqslant \sum_{0<i\leqslant q'}\Pr[C_i|\bar{C}_{i-1}] \leqslant \sum_{0\leqslant i\leqslant q'}\frac{i+n-1}{2^\ell - i - n + 1} \tag{10}$$

Now, let $i' = i + n - 1$. Then

$$\Pr[C_{q'}] \leqslant \sum_{n-1\leqslant i'\leqslant q'+n-1}\frac{i'}{2^\ell - i'} \leqslant \sum_{0\leqslant i'<q}\frac{i'}{2^\ell - q} = \frac{q(q-1)}{2\cdot(2^\ell - q)} \tag{11}$$

Finally, let R be the event that the adversary returned 1 at the end of the game – indicating a guess of 'result'. Then, noting as we have, that $\Pr_R[C_{q'}] = \Pr_G[C_{q'}]$, we get this:

$$\mathbf{Adv}^{\text{rog-cpa-}W}_{\mathcal{E}\text{-}CBCH^{P,c,n}}(A) = \Pr_R[R] - \Pr_G[R] \tag{12}$$

$$\begin{aligned} &= (\Pr_R[R|C_{q'}]\Pr_R[C_{q'}] + \Pr_R[R|\bar{C}_{q'}]\Pr_R[\bar{C}_{q'}]) - \\ &\qquad (\Pr_G[R|C_{q'}]\Pr_R[C_{q'}] + \Pr_G[R|\bar{C}_{q'}]\Pr_G[\bar{C}_{q'}]) \end{aligned} \tag{13}$$

$$= \Pr_R[R|C_{q'}]\Pr_R[C_{q'}] - \Pr_G[R|C_{q'}]\Pr_G[C_{q'}] \tag{14}$$

$$\leqslant \Pr[C_{q'}] \leqslant \frac{q(q-1)}{2\cdot(2^\ell - q)} \tag{15}$$

And we're done! □

4 Ciphertext feedback (CFB) encryption

4.1 Description

Suppose F is an ℓ-bit-to-L-bit pseudorandom function, and let $t \leqslant L$. CFB mode works as follows. Given a message X, we divide it into t-bit blocks x_0, x_1, …, x_{n-1}. Choose an initialization vector $v \in \Sigma^\ell$. We maintain a *shift register* s_i, whose initial value is v. To encrypt a block x_i, we XOR it with the result of passing the shift register through the PRF, forming y_i, and then update the shift register by shifting in the ciphertext block y_i. That is,

$$s_0 = v \qquad y_i = x_i \oplus F_K(s_i) \qquad s_{i+1} = s_i \lll_t y_i \text{ (for } 0 \leqslant i < n). \tag{16}$$

Decryption follows from noting that $x_i = y_i \oplus F_K(s_i)$. See figure 4.1 for a diagrammatic representation.

Also, we observe that the final plaintext block needn't be t bits long: we can pad it out to t bits and truncate the result without affecting our ability to decrypt.

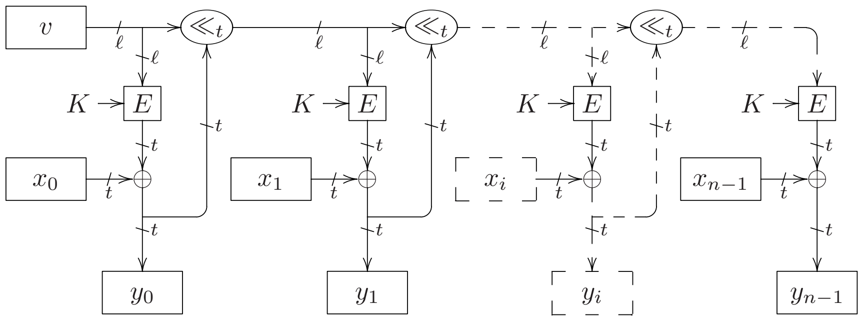

Figure 4.1: Encryption using CFB mode

4.1.1 Definition (CFB algorithms) For any function $F\colon \Sigma^\ell \to \Sigma^t$, any initialization vector $v \in \Sigma^\ell$, any plaintext $x \in \Sigma^*$ and any ciphertext $y \in \Sigma^*$, we define PRF encryption mode *CFB* $=$ (*cfb-encrypt*, *cfb-decrypt*) as follows:

Algorithm *cfb-encrypt*(F, v, x):	Algorithm *cfb-decrypt*(F, v, y):
$s \leftarrow v$;	$s \leftarrow v$;
$L \leftarrow \lvert x \rvert$;	$L \leftarrow \lvert y \rvert$;
$x \leftarrow x \parallel 0^{t\lceil L/t\rceil - L}$;	$y \leftarrow y \parallel 0^{t\lceil L/t\rceil - L}$;
$y \leftarrow \lambda$;	$x \leftarrow \lambda$;
for $i = 0$ **to** $(\lvert x \rvert - t')/t$ **do**	**for** $i = 0$ **to** $(\lvert x \rvert - t')/t$ **do**
$\quad x_i \leftarrow x[ti \mathinner{..} t(i+1)]$;	$\quad y_i \leftarrow y[ti \mathinner{..} t(i+1)]$;
$\quad y_i \leftarrow x_i \oplus F(s)$;	$\quad x_i \leftarrow x_i \oplus F(s)$;
$\quad s \leftarrow s \lll_t y_i$;	$\quad s \leftarrow s \lll_t y_i$;
$\quad y \leftarrow y \parallel y_i$;	$\quad x \leftarrow x \parallel x_i$;
return $(s, y[0 \mathinner{..} L])$;	**return** $x[0 \mathinner{..} L]$;

We now define the schemes $\mathcal{E}$-*CFB*$\$^F$, $\mathcal{E}$-*CFBC*^{F,c}, $\mathcal{E}$-*CFBE*^{F,c}, and $\mathcal{E}$-*CFBL*^{F,V_0} according to definition 2.6.4; and we define the hybrid scheme $\mathcal{E}$-*CFBH*$^{F,V_0,c}_{n_L,n_C,n_E}$ according to definition 2.6.6. □

4.2 Sliding strings

Consider for a moment the mode CFBL, i.e., with carry-over of IV from one plaintext to the next, with $t < \ell$. Then we find that some IVs are weak.

Pretend for a moment that we're an adversary playing the LOR-CPA game using an ideal random function $F \in_R \mathcal{F}^{\ell,t}$, and that the initial IV $V_0 = 0^\ell$. We choose two distinct 8-bit plaintexts l and r as our first left-or-right query. With probability 2^{-t}, the result of encrypting that first query is 0^t. However, in this case, the IV for the *next* query is $V_0 \lll_t 0^t = 0^\ell = V_0$. If this happens, we have only to submit the pair (l, l) as our second query. If the ciphertext to this second query also comes back zero, we guess that we're dealing with a left oracle; otherwise we guess right. If we don't get lucky with our first query, we just guess randomly.

This attack is shown more formally as adversary S in figure 4.2. Its resource usage is almost trivial – negligible computation and at most two encryption queries. However, its advantage

```
Adversary S^{E(·,·)}:
  l ← 0^t; r ← 0^{t−1}1;
  y ← E(l, r);
  if y[ℓ .. ℓ + t] = 0^t then
    if E(l, l) = y then b ← 0 else b ← 1;
  else
    b ←R {0, 1};
  return b;
```

Figure 4.2: Adversary S attacking $\mathcal{E}$-CFBL$^{\mathcal{F}^{\ell,t},0^\ell}$

is quite good:

$$\mathbf{Adv}^{\text{LOR-CPA}}_{\mathcal{E}\text{-CFBL}^{\mathcal{F}^{\ell,t},0^\ell}}(S) = \frac{1}{2^t}\left(1 - \frac{1}{2^t}\right).$$

This attack works because $V_0[t \mathinner{..} \ell] = V_0[0 \mathinner{..} \ell - t]$. There are similar attacks for other such relationships. The following definition characterizes these kinds of 'bad' IVs.

4.2.1 Definition (Sliding strings) We say that an ℓ-bit string x *t-slides* if there exist integers i and j such that $0 \leqslant j < i < \ell/t$ and $x[it \mathinner{..} \ell] = x[jt \mathinner{..} \ell - (i-j)t]$. □

4.2.2 Remark For all $\ell > 0$ and $t < \ell$, the string $0^{\ell-1}1$ does not t-slide. □

Alkassar, Geraldy, Pfitzmann and Sadeghi [AGPS01] offer a proof of security for CFB mode. However, they prove security only for a random initialization vector. There is a cryptic note:

> We do this for the case with random initialization vector IV. (The difference in the other case is negligible).

As we have shown above, some non-random initialization vectors lead to much decreased security. Their proof entirely ignores this possibility.

(Their result is a factor of two better than ours. We've lost this factor in passing from ROG-CPA to LOR-CPA. One can prove LOR-CPA security directly in more-or-less the same way as section 4.4 and not lose this factor.)

4.3 Security of CFB mode

4.3.1 Theorem (Security of CFB mode) *Let F be a pseudorandom function from Σ^ℓ to Σ^t; let $V_0 \in \Sigma^\ell$ be a non-t-sliding string; let c be a generalized counter in Σ^ℓ; and let n_L, n_C, n_E and q_E be nonnegative integers; and furthermore suppose that*

- $n_L + n_C + n_E \leqslant q_E$*,*
- $n_L = 0$*, or* $n_C = n_E = 0$*, or* $\ell \leqslant t$ *and* $V_0 \neq c(i)$ *for any* $n_L \leqslant i < n_L + n_C + n_E$*, and*
- *either* $n_C = 0$ *or* $\ell \leqslant t$*.*

Then, for any t_E and μ_E, and whenever we have

$$\mathbf{InSec}^{\text{lor-cpa}}(\mathcal{E}\text{-}CFBH^{F,V_0,c}_{n_L,n_C,n_E}; t_E, q_E, \mu_E) \leqslant 2 \cdot \mathbf{InSec}^{\text{prf}}(F; t_E + qt_F, q) + \frac{q(q-1)}{2^\ell}$$

where $q = \lfloor(\mu_E + q_E(t-1))/t\rfloor + n_E$, and t_F is some small constant.

The proof is a bit involved; we postpone it until section 4.4.

4.3.2 Corollary *Let F, c and V_0 be as in theorem 4.3.1. Then for any t_E, q_E and μ_E,*

$$\mathbf{InSec}^{\text{lor-cpa}}(\mathcal{E}\text{-}CFB\$^F; t_E, q_E, \mu_E) \leqslant 2 \cdot \mathbf{InSec}^{\text{prf}}(F; t_E + qt_F, q) + \frac{q(q-1)}{2^\ell}$$
$$\mathbf{InSec}^{\text{lor-cpa}}(\mathcal{E}\text{-}CFBE^{F,c}; t_E, q_E, \mu_E) \leqslant 2 \cdot \mathbf{InSec}^{\text{prf}}(F; t_E + q't_F, q') + \frac{q'(q'-1)}{2^\ell}$$
$$\mathbf{InSec}^{\text{lor-cpa}}(\mathcal{E}\text{-}CFBL^{F,V_0}; t_E, q_E, \mu_E) \leqslant 2 \cdot \mathbf{InSec}^{\text{prf}}(F; t_E + qt_F, q) + \frac{q(q-1)}{2^\ell}$$

and, if $\ell \leqslant t$,

$$\mathbf{InSec}^{\text{lor-cpa}}(\mathcal{E}\text{-}CFBC^{F,c}; t_E, q_E, \mu_E) \leqslant 2 \cdot \mathbf{InSec}^{\text{prf}}(F; t_E + qt_F, q) + \frac{q(q-1)}{2^\ell}$$

where $q = \lfloor(\mu_E + q_E(t-1))/t\rfloor + n_E$, $q' = q + q_E$, and t_F is some small constant.

Proof Follows from theorem 4.3.1 and proposition 2.6.7. □

4.3.3 Corollary *Let P be a pseudorandom permutation on Σ^ℓ, and let c and V_0 be as in theorem 4.3.1. Then for any t_E, q_E and μ_E,*

$$\mathbf{InSec}^{\text{lor-cpa}}(\mathcal{E}\text{-}CFB\$^P; t_E, q_E, \mu_E) \leqslant 2 \cdot \mathbf{InSec}^{\text{prp}}(P; t_E + qt_F, q) + \frac{q(q-1)}{2^{\ell-1}}$$
$$\mathbf{InSec}^{\text{lor-cpa}}(\mathcal{E}\text{-}CFBE^{P,c}; t_E, q_E, \mu_E) \leqslant 2 \cdot \mathbf{InSec}^{\text{prp}}(P; t_E + q't_F, q') + \frac{q'(q'-1)}{2^{\ell-1}}$$
$$\mathbf{InSec}^{\text{lor-cpa}}(\mathcal{E}\text{-}CFBL^{P,V_0}; t_E, q_E, \mu_E) \leqslant 2 \cdot \mathbf{InSec}^{\text{prp}}(P; t_E + qt_F, q) + \frac{q(q-1)}{2^{\ell-1}}$$

and, if $\ell \leqslant t$,

$$\mathbf{InSec}^{\text{lor-cpa}}(\mathcal{E}\text{-}CFBC^{P,c}; t_E, q_E, \mu_E) \leqslant 2 \cdot \mathbf{InSec}^{\text{prp}}(P; t_E + qt_F, q) + \frac{q(q-1)}{2^{\ell-1}}$$

where $q = \lfloor(\mu_E + q_E(t-1))/t\rfloor + n_E$, $q' = q + q_E$, and t_F is some small constant.

Proof Follows from corollary 4.3.2 and proposition 2.4.3. □

4.4 Proof of theorem 4.3.1

Our proof follows the same lines as for CBC mode: we show the ROG-CPA security of hybrid-CFB mode using an ideal random function, and then apply our earlier results to complete the proof. However, the ROG-CPA result will be useful later when we consider the security of OFB mode, so we shall be a little more formal about defining it.

The garbage emitter is in some sense the 'perfect' one: it emits a 'correct' IV followed by a uniform random string of the correct length.

4.4.1 Definition (The $W_\$$ garbage emitter) Let natural numbers n_L, n_C, and $V_0 \in \Sigma^\ell$ be given; then we define the garbage emitter $W_\$$ as follows.

Initialization:
$i \leftarrow 0$;
$v \leftarrow V_0$;

Garbage emitter $W_\$(m)$:
if $i < n_L$ **then** $v' \leftarrow v$;
else if $n_L \leqslant i < n_L + n_C$ **then** $v' \leftarrow c(i)$;
else if $n_L + n_C \leqslant i$ **then** $v' \xleftarrow{R} \Sigma^\ell$;
$i \leftarrow i + 1$;
$m' \leftarrow t\lfloor(m + t - 1)/t\rfloor$;
$y \xleftarrow{R} \Sigma^{m'}$;
$v \leftarrow v' \ll_{m'} y$;
return $(v', y[0 \mathinner{..} m])$ □

We now show that CFB mode with a random function is hard to distinguish from $W_\$$.

4.4.2 Lemma (Pseudorandomness of CFB mode) *Let ℓ, t, n_L, n_C, n_E, q_E, c, V_0, and q be as in theorem 4.3.1. Then, for any t_E and μ_E,*

$$\mathbf{InSec}^{\text{rog-cpa-}W_\$}(\mathcal{E}\text{-}CFBH^{\mathcal{F}^{\ell,t},V_0,c}_{n_L,n_C,n_E}; t, q_E, \mu_E) \leqslant \frac{q(q-1)}{2^{\ell+1}}.$$

Theorem 4.3.1 follows from this result by application of propositions 2.5.4 and 2.6.5. It remains therefore for us to prove lemma 4.4.2.

To reduce the weight of notation, let us agree to suppress the adornments on **Adv** and **InSec** symbols. Also, let $m_L = n_L$; let $m_C = n_L + n_C$; and let $m_E = n_L + n_C + n_E$. (Remember: the ms are cu$m$ulative.)

The truncation of ciphertext blocks makes matters complicated. Let us say that an adversary is *block-respecting* if all of its plaintext queries are a multiple of t bits in length; obviously all of the oracle responses for a block-respecting adversary are also a multiple of t bits in length.

Claim *Let A' be a block-respecting adversary querying a total of μ_E bits of plaintext queries; then*

$$\mathbf{Adv}(A') \leqslant \frac{q(q-1)}{2^{\ell+1}}$$

where $q = \mu_E/t$.

Lemma 4.4.2 follows from this claim: if A is any adversary, then we construct a block-respecting adversary A' by padding A's plaintext queries and truncating the oracle responses; and if A makes q_E queries totalling μ_E bits, then the total bits queried by A' is no more than $\lfloor(\mu_E + q_E(t-1))\rfloor$ bits. We now proceed to the proof of the above claim.

Suppose, then, that we are given a block-respecting adversary A which makes q queries to its encryption oracle. Let $F(\cdot)$ denote the application of the random function. We want to show that, provided all of the F-inputs are distinct, the F-outputs are uniformly random, and hence the CFB ciphertexts are uniformly random. As for the CBC case, life isn't that good to us: we have to deal with the case where the adversary can see that two F-inputs would have collided, and therefore that a garbage string couldn't have been generated by CFB encryption of his plaintext.

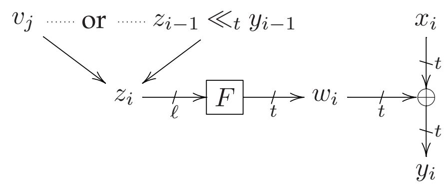

Figure 4.3: Notation for the proof of lemma 4.4.2.

Our notation will be similar to, yet slightly different from, that of section 3.4.

Let $q' = q - n_E$ be the number of t-bit plaintext blocks the adversary submits, and for $0 \leqslant i < q'$, let x_i be the ith plaintext block queried, and let y_i be the ith ciphertext block returned.

For $m_L \leqslant i < m_E$, let $c_i = c(i)$ be the ith counter value. For $0 \leqslant i < q_E$ let v_i be the ith initialization vector, i.e.,

$$v_i = \begin{cases} V_0 & \text{if } i = 0 \text{ and } n_L > 0; \\ v_{i-1} \lll_t Y_{i-1} & \text{if } 1 \leqslant i < m_L \text{ and } Y_{i-1} \text{ was the ciphertext from query } i-1; \\ c_i & \text{if } m_L \leqslant i < m_C; \\ F(c_i) & \text{if the oracle is `result', and } m_C \leqslant i < m_E\text{; or} \\ R_i & \text{for some } R_i \in_R \Sigma^\ell\text{, otherwise.} \end{cases}$$

Note that the only difference in the v_i between the 'result' and 'garbage' games occurs in the encrypted-counters phase. Furthermore, if no other F-input is equal to any c_i for $m_C \leqslant i < m_E$ then the IVs are identically distributed.

Now, for $0 \leqslant i < q'$, define

$$z_i = \begin{cases} v_j & \text{if block } i \text{ is the first block of the } j\text{th query, or} \\ z_{i-1} \lll_t y_{i-1} & \text{otherwise} \end{cases}$$

and let $w_i = x_i \oplus y_i$. In the 'result' game, we have $w_i = F(z_i)$, of course. All of this notation is summarized diagrammatically in figure 4.3. The F-inputs are precisely the z_i and c_i for $m_C \leqslant i < m_E$.

We'll denote probabilities in the 'result' game as $\Pr_R[\cdot]$ and in the 'garbage' game as $\Pr_G[\cdot]$.

Let C_r be the event, in either game, that $z_i = z_j$ for some $0 \leqslant i < j < r$, or that $z_i = c_j$ for some $0 \leqslant i < r$ and some $m_C \leqslant j < m_E$.

Let's assume that C_r didn't happen; we want the probability that C_{r+1} did, which is just the probability that z_r collides with some z_i where $0 \leqslant i < r$, or some c_i for $m_C \leqslant i < m_E$. Observe that, under this assumption, all the w_i, and hence the y_i, are uniformly distributed, and that therefore the two games are indistinguishable.

One of the following cases holds.

1. If $r = 0$ and $m_L > 0$ then $z_r = V_0$. There is no other z_i yet for z_r to collide with, though it might collide with some encrypted counter $F(c_i)$, with probability $n_E/2^\ell$.

2. If $z_r = c_i$ is the IV for some message i where $m_L \leqslant i < m_C$, life is a bit complicated. It can't collide with V_0 or other c_i by assumption; the encrypted counters and random

IVs haven't been chosen yet; and either $n_C = 0$ (in which case there's nothing to do here anyway) or $\ell \leqslant t$, so there are no z_i containing partial copies of V_0 to worry about. This leaves non-IV z_i: again, $\ell \leqslant t$, so $z_i = y_i[t - \ell \mathinner{..} t]$, which is random by our assumption of $\bar{C}_r$; hence a collision with one of these z_i occurs with probability at most $r/2^\ell$.

3. If z_r is the IV for some message i where $m_C \leqslant i < m_E$, then it can collide with previous z_i or either previous or future c_i. We know, however, that no F-input has collided with c_i, so in the 'result' game, $z_r = F(c_r)$ is uniformly distributed; in the 'garbage' game, $W_\$$ generates z_r at random anyway. It collides, therefore, with probability at most $(r + n_E)/2^\ell$.

4. If z_r is the IV for some message i where $m_E \leqslant i < q'$ then z_r was chosen uniformly at random. Hence it collides with probability at most $(r + n_E)/2^\ell$.

5. Finally, either z_r is not the IV for a message, or it is, but the message number $i < n_L$, so in either case, $z_r = z_{r-1} \ll_t y_{r-1}$. We have two subcases to consider.

   (a) If $1 \leqslant r < \ell/t$ (we dealt with the case $r = 0$ above) then some of V_0 remains in the shift register. If z_r collides with some z_i, for $0 \leqslant i < r$, then we must have $z_r[0 \mathinner{..} \ell - tr] = z_i[0 \mathinner{..} \ell - tr]$; but $z_r[0 \mathinner{..} \ell - tr] = V_0[tr \mathinner{..} \ell]$, and $z_i[0 \mathinner{..} \ell - tr] = V_0[ti \mathinner{..} \ell - t(r-i)]$, i.e., we have found a t-sliding of V_0, which is impossible by hypothesis. Hence, z_r cannot collide with any earlier z_i. Also by hypothesis, $n_C = n_E = 0$ if $\ell > t$, so z_r cannot collide with any counters c_i.

   (b) Suppose, then, that $r \geqslant \ell/t$. For $0 \leqslant j < \ell/t$, let $H_j = \ell - tj$, $L_j = \max(0, H_j - t)$, and $N_j = H_j - L_j$. (Note that $\sum_{0 \leqslant j < \ell/t} N_j = \ell$.) Then $z_r[L_j \mathinner{..} H_j] = y_{r-j-1}[t - N_j \mathinner{..} t]$; but the y_i for $i < r$ are uniformly distributed. Thus, z_r collides with some specific other value z' only with probability $1/2^{\sum_j N_j} = 1/2^\ell$. The overall collision probability for z_r is then at most $(r + n_E)/2^\ell$.

In all these cases, it's clear that the collision probability is no more than $(r + n_E)/2^\ell$.

The probability that there is a collision during the course of the game is $\Pr[C_{q'}]$, which we can now bound thus:

$$\Pr[C'_q] \leqslant \sum_{0 < i \leqslant q'} \Pr[C_i | \bar{C}_{i-1}] \leqslant \sum_{0 < i \leqslant q'} \frac{i + n_E}{2^\ell}. \tag{17}$$

If we set $i' = i + n_E$, then we get

$$\Pr[C'_q] \leqslant \sum_{0 \leqslant i' \leqslant q} \frac{i'}{2^\ell} = \frac{q(q-1)}{2^{\ell+1}}. \tag{18}$$

Finally, then, we can apply the same argument as we used at the end of section 3.4 to show that

$$\mathbf{Adv}(A') \leqslant \frac{q(q-1)}{2^{\ell+1}} \tag{19}$$

as claimed. This completes the proof.

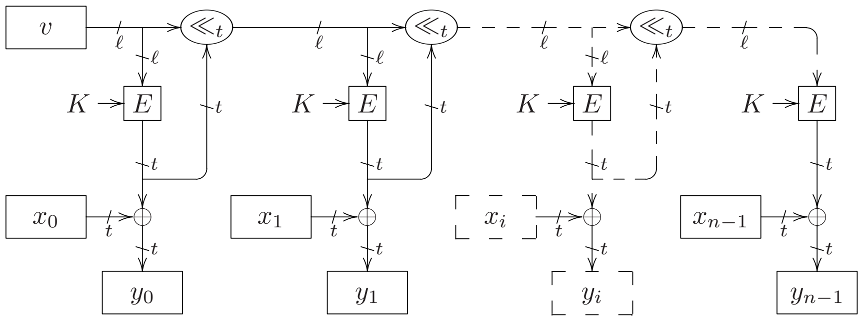

Figure 5.1: Encryption using OFB mode

5 OFB mode encryption

5.1 Description

Suppose F is an ℓ-bit-to-L-bit pseudorandom function, and let $t \leqslant L$. OFB mode works as follows. Given a message X, we divide it into t-bit blocks x_0, x_1, …, x_{n-1}. Choose an initialization vector $v \in \Sigma^\ell$. We maintain a *shift register* s_i, whose initial value is v. To encrypt a block x_i, we XOR it with the result z_i of passing the shift register through the PRF, forming y_i, and then update the shift register by shifting in the PRF output z_i. That is,

$$s_0 = v \qquad z_i = F_K(s_i) \qquad y_i = x_i \oplus z_i \qquad s_{i+1} = s_i \ll_t z_i \text{ (for } 0 \leqslant i < n). \tag{20}$$

Decryption is precisely the same operation.

Also, we observe that the final plaintext block needn't be t bits long: we can pad it out to t bits and truncate the result without affecting our ability to decrypt.

5.1.1 Definition (OFB algorithms) For any function $F\colon \Sigma^\ell \to \Sigma^t$, any initialization vector $v \in \Sigma^\ell$, any plaintext $x \in \Sigma^*$ and any ciphertext $y \in \Sigma^*$, we define PRF encryption mode $\mathit{OFB} = (\mathit{ofb\text{-}encrypt}, \mathit{ofb\text{-}decrypt})$ as follows:

Algorithm *ofb-encrypt*(F, v, x):
  $s \leftarrow v$;
  $L \leftarrow |x|$;
  $x \leftarrow x \parallel 0^{t\lceil L/t\rceil - L}$;
  $y \leftarrow \lambda$;
  for $i = 0$ **to** $(|x| - t')/t$ **do**
    $x_i \leftarrow x[ti \,..\, t(i+1)]$;
    $z_i \leftarrow F(s)$;
    $y_i \leftarrow x_i \oplus z_i$;
    $s \leftarrow s \ll_t z_i$;
    $y \leftarrow y \parallel y_i$;
  return $(s, y[0 \,..\, L])$;

Algorithm *ofb-decrypt*(F, v, y):
  return *ofb-encrypt*(F, v, y);

We now define the schemes $\mathcal{E}\text{-}\mathit{OFB\$}^F$, $\mathcal{E}\text{-}\mathit{OFBC}^{F,c}$, $\mathcal{E}\text{-}\mathit{OFBE}^{F,c}$, and $\mathcal{E}\text{-}\mathit{OFBL}^{F,V_0}$ according to definition 2.6.4; and we define the hybrid scheme $\mathcal{E}\text{-}\mathit{OFBH}^{F,V_0,c}_{n_L,n_C,n_E}$ according to definition 2.6.6. □

5.1.2 Remark (Similarity to CFB mode) OFB mode is strongly related to CFB mode: we can OFB encrypt a message x by *CFB-encrypting* the all-zero string $0^{|x|}$ with the same key and IV. That is, we could have written *ofb-encrypt* and *ofb-decrypt* like this:

Algorithm *ofb-encrypt*(F, v, x): $(s, z) \leftarrow$ *cfb-encrypt*$(F, v, 0^{|x|})$; **return** $(s, x \oplus z)$;	Algorithm *ofb-decrypt*(F, v, y): **return** *ofb-encrypt*(F, v, y);

We shall use this fact to prove the security of OFB mode in the next section. □

5.2 Security of OFB mode

5.2.1 Theorem (Security of OFB mode) *Let F be a pseudorandom function from Σ^ℓ to Σ^t; let $V_0 \in \Sigma^\ell$ be a non-t-sliding string; let c be a generalized counter in Σ^ℓ; and let n_L, n_C, n_E and q_E be nonnegative integers; and furthermore suppose that*

- $n_L + n_C + n_E \leqslant q_E$,
- $n_L = 0$, *or* $n_C = n_E = 0$, *or* $\ell \leqslant t$ *and* $V_0 \neq c(i)$ *for any* $n_L \leqslant i < n_L + n_C + n_E$, *and*
- *either* $n_C = 0$ *or* $\ell \leqslant t$.

Then, for any t_E and μ_E, and whenever we have

$$\mathbf{InSec}^{\text{lor-cpa}}(\mathcal{E}\text{-}\mathit{OFBH}^{F,V_0,c}_{n_L,n_C,n_E}; t_E, q_E, \mu_E) \leqslant 2 \cdot \mathbf{InSec}^{\text{prf}}(F; t_E + q t_F, q) + \frac{q(q-1)}{2^\ell}$$

where $q = \lfloor (\mu_E + q_E(t-1))/t \rfloor + n_E$, and t_F is some small constant.

Proof We claim that

$$\mathbf{InSec}^{\text{rog-cpa-}W_\$}(\mathcal{E}\text{-}\mathit{OFBH}^{\mathcal{F}^{\ell,t},V_0,c}_{n_L,n_C,n_E}; t, q_E, \mu_E) \leqslant \frac{q(q-1)}{2^{\ell+1}}.$$

This follows from lemma 4.4.2, which makes the same statement about CFB mode, and the observation in remark 5.1.2. Suppose A attempts to distinguish OFBH encryption from $W_\$$. We define the adversary B which uses A to attack CFBH encryption, as follows:

Adversary $B^{E(\cdot)}$: **return** $A^{\mathit{ofb}(\cdot)}$;	Function *ofb*(x): $(v, z) \leftarrow E(0^{|x|})$; **return** $(v, x \oplus z)$;

Now we apply proposition 2.5.4; the theorem follows. □

5.2.2 Corollary *Let F, c and V_0 be as in theorem 5.2.1. Then for any t_E, q_E and μ_E,*

$$\mathbf{InSec}^{\text{lor-cpa}}(\mathcal{E}\text{-}\mathit{OFB\$}^F; t_E, q_E, \mu_E) \leqslant 2 \cdot \mathbf{InSec}^{\text{prf}}(F; t_E + q t_F, q) + \frac{q(q-1)}{2^\ell}$$

$$\mathbf{InSec}^{\text{lor-cpa}}(\mathcal{E}\text{-}\mathit{OFBE}^{F,c}; t_E, q_E, \mu_E) \leqslant 2 \cdot \mathbf{InSec}^{\text{prf}}(F; t_E + q' t_F, q') + \frac{q'(q'-1)}{2^\ell}$$

$$\mathbf{InSec}^{\text{lor-cpa}}(\mathcal{E}\text{-}\mathit{OFBL}^{F,V_0}; t_E, q_E, \mu_E) \leqslant 2 \cdot \mathbf{InSec}^{\text{prf}}(F; t_E + q t_F, q) + \frac{q(q-1)}{2^\ell}$$

and, if $\ell \leqslant t$,

$$\mathbf{InSec}^{\text{lor-cpa}}(\mathcal{E}\text{-}\mathit{OFBC}^{F,c}; t_E, q_E, \mu_E) \leqslant 2 \cdot \mathbf{InSec}^{\text{prf}}(F; t_E + q t_F, q) + \frac{q(q-1)}{2^\ell}$$

where $q = \lfloor (\mu_E + q_E(t-1))/t \rfloor + n_E$, $q' = q + q_E$, and t_F is some small constant.

Proof Follows from theorem 5.2.1 and proposition 2.6.7. □

5.2.3 Corollary *Let P be a pseudorandom permutation on Σ^ℓ, and let c and V_0 be as in theorem 5.2.1. Then for any t_E, q_E and μ_E,*

$$\mathbf{InSec}^{\text{lor-cpa}}(\mathcal{E}\text{-}\mathit{OFB}\$^P; t_E, q_E, \mu_E) \leqslant 2 \cdot \mathbf{InSec}^{\text{prp}}(P; t_E + qt_F, q) + \frac{q(q-1)}{2^\ell}$$
$$\mathbf{InSec}^{\text{lor-cpa}}(\mathcal{E}\text{-}\mathit{OFBE}^{P,c}; t_E, q_E, \mu_E) \leqslant 2 \cdot \mathbf{InSec}^{\text{prp}}(P; t_E + q't_F, q') + \frac{q'(q'-1)}{2^\ell}$$
$$\mathbf{InSec}^{\text{lor-cpa}}(\mathcal{E}\text{-}\mathit{OFBL}^{P,V_0}; t_E, q_E, \mu_E) \leqslant 2 \cdot \mathbf{InSec}^{\text{prp}}(P; t_E + qt_F, q) + \frac{q(q-1)}{2^\ell}$$

and, if $\ell \leqslant t$,

$$\mathbf{InSec}^{\text{lor-cpa}}(\mathcal{E}\text{-}\mathit{OFBC}^{P,c}; t_E, q_E, \mu_E) \leqslant 2 \cdot \mathbf{InSec}^{\text{prp}}(P; t_E + qt_F, q) + \frac{q(q-1)}{2^\ell}$$

where $q = \lfloor (\mu_E + q_E(t-1))/t \rfloor + n_E$, $q' = q + q_E$, and t_F is some small constant.

Proof Follows from corollary 5.2.2 and proposition 2.4.3. □

6 Acknowledgements

Thanks are due to David Wagner for pointing me at [AGPS01] and warning me of the dangers of sliding IVs in CFB mode. Thanks also to Clive Jones for his suggestions on notation, and his help in structuring the proofs.

7 References

[AGPS01] Ammar Alkassar, Alexander Geraldy, Birgit Pfitzmann, and Ahmad-Reza Sadeghi; *Optimized self-synchronizing mode of operation*; in Mitsuru Matsui, ed., *FSE*; vol. 2355 of *Lecture Notes in Computer Science*; Springer; 2001; ISBN 3-540-43869-6; URL `http://citeseer.nj.nec.com/alkassar01optimized.html`.

[BDJR97] Mihir Bellare, Anand Desai, E. Jokipii, and Phillip Rogaway; *A concrete security treatment of symmetric encryption*; in *IEEE Symposium on Foundations of Computer Science*; 1997; pp. 394–403; URL `http://www-cse.ucsd.edu/users/mihir/papers/sym-enc.html`.

[BKR94] Mihir Bellare, Joe Kilian, and Phillip Rogaway; *The security of cipher block chaining*; in Yvo G. Desmedt, ed., *Advances in cryptology, CRYPTO '94: 14th annual international cryptology conference, Santa Barbara, California, USA, August 21–25, 1994: proceedings*; vol. 839 of *Lecture Notes in Computer Science*; Springer-Verlag, Berlin, Germany / Heidelberg, Germany / London, UK / etc.; 1994; ISBN 3-540-58333-5 (Berlin), 0-387-58333-5 (New York); ISSN 0302-9743; pp. 341–358.

[BR96] R. Baldwin and R. Rivest; *RFC 2040: The RC5, RC5-CBC, RC5-CBC-pad, and RC5-CTS algorithms*; October 1996; URL `ftp://ftp.internic.net/rfc/rfc2040.txt`, `ftp://ftp.math.utah.edu/pub/rfc/rfc2040.txt`; status: INFORMATIONAL.

[Dae95] Joan Daemen; *Cipher and hash function design strategies based on linear and differential cryptanalysis*; Ph.D. thesis; K. U. Leuven; 1995.

[RBBK01] Phillip Rogaway, Mihir Bellare, John Black, and Ted Krovetz; *OCB: a block-cipher mode of operation for efficient authenticated encryption*; in *ACM Conference on Computer and Communications Security*; 2001; pp. 196–205; URL http://www.cs.ucdavis.edu/~rogaway/ocb/.

[Sch96] Bruce Schneier; *Applied Cryptography: Protocols, Algorithms, and Source Code in C*; John Wiley and Sons, Inc., New York, NY, USA; Second edn.; 1996; ISBN 0-471-12845-7 (cloth), 0-471-11709-9 (paper); URL http://www.counterpane.com/applied.html.

[Uni81] United States. National Bureau of Standards; *FIPS pub 81: DES modes of operation*; December 1981.