SCB Mode: Semantically Secure Length-Preserving Encryption

Fabio Banfi

Department of Computer Science
ETH Zurich
8092 Zurich, Switzerland
fabio.banfi@inf.ethz.ch

Abstract. To achieve semantic security, symmetric encryption schemes classically require ciphertext expansion. In this paper we provide a means to achieve semantic security while preserving the length of messages at the cost of mildly sacrificing correctness. Concretely, we propose a new scheme that can be interpreted as a secure alternative to (or wrapper around) plain Electronic Codebook (ECB) mode of encryption, and for this reason we name it Secure Codebook (SCB). Our scheme is the first length-preserving encryption scheme to effectively achieve semantic security.

Keywords: SCB · secure codebook · semantic security · length-preserving encryption

1 Introduction

In this paper we revisit the classical insecure Electronic Codebook (ECB) mode of encryption, and transform it into one that achieves semantic security, denoted Secure Codebook (SCB). This will be at the cost of imperfect correctness and the need of keeping state, but we provide optimal security-correctness trade-offs depending on the setting.

1.1 Background and Motivation

Given a block cipher over $\{0,1\}^n$, typical modes of operation for symmetric encryption map plaintext messages of length ℓn to ciphertexts of length at least $(\ell+1)n$. An exception is the Electronic Codebook (ECB) mode of operation, whose ciphertext have length exactly ℓn, and is therefore *length-preserving*. Still, ECB is the archetypal example of an insecure mode of encryption, in that it fails to achieve semantic security. The reason is that the latter, conventionally phrased in terms of *indistinguishability under a chosen-plaintext attack* (IND-CPA), is easily broken for ECB simply by comparing the encryption of a message with repeating blocks against one without (or a uniformly chosen bit string of the same length).

Semantic security has been initially introduced and achieved by Goldwasser and Micali [GM82], where it was originally defined for *probabilistic* (public-key) encryption. Bellare et al. [BDJR97] later adapted the notion to symmetric encryption, and subsequently Rogaway [Rog04] initiated the rigorous study of semantically secure *deterministic* (symmetric) encryption, made possible by the use of *nonces*. A typical example of a semantically secure mode of encryption is counter (CTR) mode, where a nonce of n bits (or simply a random string, if one wants to obtain a probabilistic encryption scheme) is constantly increased and fed to the block cipher to obtain pseudorandom bit strings to be used as (one-time) pads for each message block. By virtue of the nonce being used as a counter and thus never repeating, and the PRP/PRF switching lemma, we are guaranteed that each block, even

repeated ones, will be padded with bit string that are computationally indistinguishable from uniformly and independently distributed ones. But clearly, the nonce needs to be transmitted (in or out of band) as part of the ciphertext, effectively contributing to the expansion of the ciphertext by one block.

Still, with CTR mode one can in principle achieve length preservation by pre-agreeing on an initial counter, and then keeping state. But crucially, it is imperative that *ciphertexts do not get reordered or dropped (deleted) in transit.* To see how this is possible, imagine that Alice and Bob agree to set the initial counter to 1, and suppose for simplicity that Alice wants to send block messages $M_1, \ldots, M_\ell \in \{0,1\}^n$ to Bob. She will encrypt each message as $C_i \doteq E_K(i) \oplus M_i$ using the secret key K and a pre-agreed block cipher E, but an adversary will reorder the ciphertexts in transit by effectively delivering the permuted sequence $C'_1, \ldots, C'_\ell$ to Bob, where $C'_i \doteq C_{\pi(i)}$, for some permutation π. This means that Bob will then decrypt each message as $M'_i \doteq E_K(i) \oplus C'_i$, but clearly $M'_i = M_i$ with probability 1 if and only if $\pi(i) = i$. Therefore, only messages that have been encrypted using a counter that is a fixed point of π will be correctly decrypted with probability 1. Moreover, in case even just one of the ciphertext is dropped by the adversary (and even if the others are not permuted), correctness of the scheme is completely lost for any subsequent message.

It is nevertheless well known that length-preserving encryption (or rather, *enciphering*) is indeed possible, if one wants to give up semantic security and settle for the weaker notion of pseudo-random permutation (PRP) security, adapted to variable length bit strings. The study of *variable-input-length* (VIL) ciphers was initiated by Bellare and Rogaway [BR99], and asks the question of how to transform a block cipher for fixed-input length into a VIL cipher, in such a way that the length is preserved. This is useful for example in networking applications where a packet format needs to be upgraded by adding privacy features under the constraint that such packet preserves its exact structure.

Bellare and Rogaway stated that *"semantically secure encryption cannot possibly be length preserving"*, and here we want to exactly challenge that statement. We argue that this is not only a theoretically interesting question, but it also captures scenarios that are not entirely unlikely in practice. For example, consider a small low-power IoT device that needs to communicate data confidentially in a strong sense (that is, in a semantically secure sense) via UDP packets to some server, knowing that plaintext messages might be repeated, or might simply contain repeating blocks within or across messages. Note that since UDP, unlike TCP, does not define a session construct, reordering and dropping (deletion) of packets cannot be excluded. Now, it is reasonable to assume that the device might need to regularly send and receive a small number of encrypted messages, say each day d sessions consisting of m messages each, and that each message is made up of a small number of blocks (whose length is defined by the block cipher), say at most b. Then, with conventional IND-CPA secure modes of encryption, the amount of overall blocks that need to be transmitted per day is upper bounded by $dmb + dm$, but if the scheme would be length-preserving, then this bound would only amount to dmb, which for large d and small m and b represents a significant gain in communication efficiency. To achieve length preservation, the above approach with CTR mode would *not* be suitable in this setting, because using UDP might cause ciphertexts to be reordered or deleted. This naturally leads us to the question that we aim at answering in this paper, namely:

> *Can we achieve semantically secure length-preserving symmetric encryption such that correctness is not completely lost in case of reordering or deletion?*

We will answer this question in the positive for the case of reordering, and show that it is indeed possible to achieve the desired goal at the cost of (tunable) imperfect correctness and keeping state.

1.2 Contribution

We answer the above question constructively, by providing a concrete stateful symmetric encryption scheme that achieves semantic (IND-CPA) security, but that has imperfect correctness. This does not mean that the scheme is impractical, but rather, that it should be used in a way that (provably) provides the desired level of correctness. The core of our scheme is a mode of encryption, that is, a way to use a block cipher $E : \{0,1\}^\kappa \times \{0,1\}^n \to \{0,1\}^n$ in order to encrypt bit strings whose length is a multiple of the block length n.

Our mode of encryption can be seen as a (black-box) adaptation of the classical (but clearly not semantically secure) Electronic Codebook (ECB) mode, and for this reason we named our new mode Secure Codebook (SCB). More precisely, given oracle access to ECB encryption and decryption functions (that is, for an unknown secret key K_1), SCB encryption uses an additional key K_2, keeps state, and uses the ECB encryption oracle, and SCB decryption also uses K_2, keeps state, and uses the ECB decryption oracle. Therefore, if for any (probably unsound) reason a protocol employs ECB as a sub-module, it is possible to wrap such module in a black-box way and achieve what any semantically secure scheme would achieve in its place, without having to change the syntax of the exchanged messages, and only at the cost of keeping some state and scarifying some correctness (still, in a way that it is very close to perfect, if the parameters are reasonably chosen).

Our approach is quite simple: Anytime a block is repeated (within or across messages), rather than enciphering it again with the block cipher, we will encipher a specially formatted block that, if decrypted, with high enough probability will be detected as signaling the repetition of a previous block. Clearly, forcing every block (even such repetition signals) to be of the same size, must be at the cost of correctness. More precisely, we will be splitting the block cipher domain in valid messages and repetition signals, so it is imperative to design such split in a careful way, so that with good enough probability deciphered blocks will not be interpreted as repetitions. Finally, applying the so-called *ciphertext stealing* (CTS) paradigm, turns our mode of encryption into an encryption scheme for strings of any length. Therefore, our scheme can also be easily adapted to be a (PRP) secure length-preserving enciphering scheme (or VIL cipher), even though, one with imperfect correctness. This can be achieved simply by *not* keeping state.

We provide two variants of our scheme. The simpler one can be used in a setting where no reordering of transmitted ciphertexts takes place. The second one is slightly more complex, because it assumes reordering of transmitted ciphertexts does take place. The idea is simply to tag blocks of decrypted messages that had the structure of a repetition, but for which no reference block has been deciphered so far. This way, if a message containing the reference block is decrypted later, the receiver can (at least partially) reconstruct the previous messages with tagged blocks. Note that this feature is impossible to achieve with CTR mode, if reordering or deletion of transmitted ciphertexts take place.

1.3 Related Work

The idea of designing length-preserving (symmetric) encryption schemes is not new. What is new is the axis used to approach this problem: Rather than asking how to appropriately weaken the security notion of such a scheme, we ask how we can weaken its correctness, while maintaining a high level of security, but in such a way that the scheme is still usable in practice. In fact, previous work on length-preserving encryption (LPE) addresses the problem we are considering here, but from a different perspective. Introduced by Bellare and Rogaway [BR99], LPE can essentially be understood as the problem of turning a regular (fixed-size) block cipher into a VIL cipher, such that for each possible length ℓ (and key), the new scheme implements a permutation over bit strings of length ℓ. Clearly, this implies that semantic security is unattainable by an LPE scheme, and therefore the best hope is to achieve a modification the classical of pseudo-random permutation (PRP)

security notion for block ciphers, which simply asks that for each possible length ℓ (and key), the scheme is indistinguishable from a uniformly random permutation over bit strings of length ℓ. For this reason, it would in fact be more correct to understand the E in LPE as *enciphering*, rather than encryption.

The original work by Bellare and Rogaway [BR99] provided a concrete scheme that can be abstractly described as a two-pass CBC-MAC over the input message of arbitrary size. Subsequently, Bleichenbacher and Desai [BD99] refined this scheme to achieve Strong PRP (SPRP) security. More efficient constructions were later found by Patel et al. [PRS04]. In [CYK04] and [CKY07], Cook et al. introduce the notion of *elastic* block ciphers (EBC), which unlike the previous works, achieves LPE by treating only the round function of the underlying block cipher as a black box, not the entire block cipher.

In the literature, there has been a shift in attention towards achieving *tweakable* LPE from a tweakable block cipher, a primitive originally introduced by Liskov et al. [LRW02]. Two first such schemes are the modes of operation CMC [HR03] and EME [HR04], both introduced by Halevi and Rogaway, but both limited to input lengths which are a multiple of the block length (hence LPE schemes that are *not* VIL schemes). The first truly VIL tweakable LPE scheme, the Extended Codebook (XCB) mode of operation, was described by McGrew and Fluhrer [MF04], who combined a block cipher with an universal hash function to realize an unbalanced three-round Feistel network. The same authors only later formally proved its security in [MF07]. Halevi extended EME into EME* [Hal04], achieving a fully VIL tweakable LPE scheme; Wang et al. [WFW05] proposed HCTR, based on the counter (CTR) mode of encryption, and later Chakraborty and Nandi [CN08] improved its security bound. A series of schemes based on ECB followed, PEP by Chakraborty and Sarkar [CS06], TET by Halevi [Hal07], and HEH by Sarkar [Sar07]. Further schemes improving on HCTR are HSE by Minematsu and Matsushima [MM07], HCH by Chakraborty and Sarkar [CS08], and HMC by Nandi [Nan08]. More recent schemes include TCT_1 and TCT_2 by Shrimpton and Terashima [ST13] and Adiantum by Crowley and Biggers [CB18].

We stress again that of all the above mentioned works, none provides an LPE scheme achieving semantic security, which is precisely what we do here for the first time.

2 Preliminaries

2.1 Notation

Let $\mathbb{N} = \{1, 2, \ldots\}$. For any $n \in \mathbb{N}$, we use the convention $[n] \doteq \{1, \ldots, n\}$. For a set S we denote the set of all (non-empty) sequences of length at least n over S as $S^{\geq n} \doteq \cup_{i \geq n} S^i$, and we also define $S^+ \doteq S^{\geq 1}$. For some $x = (x_\ell, \ldots, x_1) \in S^+$, with $\ell \in \mathbb{N}$, we define $|x| \doteq \ell$ as well as $[x]_t \doteq (x_t, \ldots, x_1)$, for any $1 \leq t \leq \ell$. For another $y = (y_{\ell'}, \ldots, y_1) \in S^+$, with $\ell' \in \mathbb{N}$, we define $x \| y \doteq (x_\ell, \ldots, x_1, y_{\ell'}, \ldots, y_1)$. When $S = \{0,1\}$, we call such sequences bit strings. For any $n \in \mathbb{N}$, by $\mathcal{F}_n$ we denote the set of all functions $\{0,1\}^n \to \{0,1\}^n$, and by $\mathcal{P}_n$ the set of all bijections (permutations) $\{0,1\}^n \to \{0,1\}^n$. For any $k, v \in \mathbb{N}$, we model a look-up table $\mathbf{T}$ mapping key bit strings of length k to value bit strings of length v as a function $\{0,1\}^k \to \{0,1\}^v \cup \{\perp\}$ (with $\perp \notin \{0,1\}^+$), and we define the following operations: Initializing a look-up table $\mathbf{T}$ to an empty one is denoted $\mathbf{T} \leftarrow [\,]$; Assigning value V to key K in $\mathbf{T}$ is denoted $\mathbf{T}[K] \leftarrow V$, and we assume that any value previously assigned to K will be overwritten by V; Reading the value assigned to key K in $\mathbf{T}$ and assigning it in V is denoted $V \leftarrow \mathbf{T}[K]$, and if $\mathbf{T}$ does not hold any value for K (that is, no value has been assigned to K in $\mathbf{T}$ before), then V will be assigned the special symbol $\perp$. Finally, if X is a finite set, we let $x \xleftarrow{\$} X$ denote picking an element of X uniformly at random and assigning it to x, and for an algorithm A we let $y \leftarrow A^{O_1, O_2, \cdots}$ denote running A with oracle access to $O_1, O_2, \ldots$, modeled as functions, and assigning the output to y.

2.2 Games, Adversaries, and Reductions

We work in the concrete security setting pioneered by Bellare et al. [BKR94, BDJR97], and use the code-based game-playing framework of Bellare and Rogaway [BR06]. A game G specifies a number of procedures $\mathrm{O}_1, \mathrm{O}_2, \ldots$ that model oracles for an adversary A. G also optionally defines a procedure Init, and (if not specified otherwise), A will output a bit b. Execution of adversary A with game G then consists of running A with oracle access to Init (if present) and $\mathrm{O}_1, \mathrm{O}_2, \ldots$, with the restrictions that A's single call to Init (if present) must be its first overall call. The output of the execution is the bit output by A, and we use the notation $\Pr[\mathrm{G}(A)] \doteq \Pr[b = 0 \mid b \leftarrow A^{\textsc{Init},\mathrm{O}_1,\mathrm{O}_2,\cdots}]$. For some security notions, when defining A's advantage, we will directly use the right-hand side expression with oracles modeled by regular functions parameterized by random variables.

2.3 Symmetric Encryption

In this work we consider a special type of symmetric encryption: *length-preserving* and *stateful*. We also restrict our attention to schemes with message and ciphertext spaces consisting of bit strings with lengths that are integer multiples of a fixed block length, not bit strings of arbitrary length. This makes our exposition easier, and our result more modular, because we will achieve schemes for *any* length by simply using ciphertext stealing as the very last step (see Section 3.4).

Definition 1. For some $n \in \mathbb{N}$, a *Length-Preserving Stateful Encryption* (LPSE) scheme $\Pi = (\mathcal{E}, \mathcal{D})$ specifies *stateful* algorithms $\mathcal{E} : \mathcal{K} \times (\{0,1\}^n)^+ \times \mathcal{S} \to (\{0,1\}^n)^+ \times \mathcal{S}$ and $\mathcal{D} : \mathcal{K} \times (\{0,1\}^n)^+ \times \mathcal{T} \to (\{0,1\}^n)^+ \times \mathcal{T}$, for some $n \in \mathbb{N}$. $\mathcal{K}$ is the space of keys, $\mathcal{S}$ is the space of encryption states, and $\mathcal{T}$ is the space of decryption states. The encryption algorithm takes as input a key $K \in \mathcal{K}$, a message $M \in \{0,1\}^{\ell n}$, for some $\ell \in \mathbb{N}$, and an encryption state $\mathbf{S} \in \mathcal{S}$, and returns a ciphertext-state pair $(C; \mathbf{S}') \leftarrow \Pi.\mathcal{E}(K, M; \mathbf{S})$, such that $C \in \{0,1\}^{\ell n}$ (length preservation). The decryption algorithm takes as input a key $K \in \mathcal{K}$, a ciphertext $C \in \{0,1\}^{\ell n}$, for some $\ell \in \mathbb{N}$, and a decryption state $\mathbf{T} \in \mathcal{T}$, and returns a message-state pair $(M; \mathbf{T}') \leftarrow \Pi.\mathcal{D}(K, C; \mathbf{T})$, such that $M \in \{0,1\}^{\ell n}$.

In this paper we will assume for simplicity that the key distribution is the uniform distribution over $\mathcal{K}$. Note that we did not include any correctness requirement in Definition 1. For better readability, we will use the following short-hand notation:

- For any key K, encryption state $\mathbf{S}$, message M, decryption state $\mathbf{T}$, and ciphertext C, we define $\mathcal{E}_K^{\mathbf{S}}(M) \doteq \mathcal{E}(K, M; \mathbf{S})$ and $\mathcal{D}_K^{\mathbf{T}}(C) \doteq \mathcal{D}(K, C; \mathbf{T})$.
- We assume that (syntactically) the state is *"passed by reference"* to the encryption and decryption algorithms, that is, we write $C \leftarrow \mathcal{E}_K^{\mathbf{S}}(M)$ to mean the sequence of operations $(C; \mathbf{S}') \leftarrow \mathcal{E}_K^{\mathbf{S}}(M)$, $\mathbf{S} \leftarrow \mathbf{S}'$, and $M \leftarrow \mathcal{D}_K^{\mathbf{T}}(C)$ to mean the sequence of operations $(M; \mathbf{T}') \leftarrow \mathcal{D}_K^{\mathbf{T}}(C)$, $\mathbf{T} \leftarrow \mathbf{T}'$, meaning that encryption and decryption algorithms *implicitly modify the state as a side effect*.

Finally, we introduce a new operation that allows to enhance correctness of an LPSE scheme in case transmitted ciphertexts are reordered (for which we already apply the analogous above remarks).

Definition 2. For some $n \in \mathbb{N}$, a *Recoverable LPSE* (R-LPSE) scheme $\Pi = (\mathcal{E}, \mathcal{D}, \widetilde{\mathcal{D}}, \mathcal{R})$ is an LPSE scheme $(\mathcal{E}, \mathcal{D})$ which additionally defines a *tagged* decryption algorithm $\widetilde{\mathcal{D}} : \mathcal{K} \times \mathcal{T} \times (\{0,1\}^n)^+ \to (\{0,1\}^n)^+ \times \{0,1\}^+$ and a *recovery* algorithm $\mathcal{R} : \mathcal{K} \times ((\{0,1\}^n)^+ \times \{0,1\}^+)^+ \to (((\{0,1\}^n)^+)^+$. The tagged decryption algorithm takes as input a key $K \in \mathcal{K}$, a decryption state $\mathbf{T} \in \mathcal{T}$, and a ciphertext $C \in \{0,1\}^{\ell n}$, for some $\ell \in \mathbb{N}$, and returns a tagged message $(M, \boldsymbol{t}) \leftarrow \Pi.\widetilde{\mathcal{D}}_K^{\mathbf{T}}(C)$, such that $M = \Pi.\mathcal{D}_K^{\mathbf{T}}(C)$

Game $\mathrm{G}_{\Pi}^{\text{ind-cpa-0}}$	Game $\mathrm{G}_{\Pi}^{\text{ind-cpa-1}}$
1 : **procedure** INIT	1 : **procedure** ENC(M)
2 : $K \xleftarrow{\$} \mathcal{K}$	2 : $C \xleftarrow{\$} \{0,1\}^{\lvert M\rvert}$
3 : $\mathbf{S} \leftarrow [\,]$	3 : **return** C
4 : **procedure** ENC(M)	
5 : $C \leftarrow \Pi.\mathcal{E}_K^{\mathbf{S}}(M)$	
6 : **return** C	

Figure 1: Games defining semantic security of an LPSE scheme Π.

and $\boldsymbol{t} \in \{0,1\}^{\ell}$. The recovery algorithm takes as input a key $K \in \mathcal{K}$ and a list of tagged messages $(M_1, \boldsymbol{t}_1), \ldots, (M_s, \boldsymbol{t}_s) \in (\{0,1\}^n)^+$, for some $s \in \mathbb{N}$, and returns a list of messages $(M'_1, \ldots, M'_s) \leftarrow \Pi.\mathcal{R}_K((M_1, \boldsymbol{t}_1), \ldots, (M_s, \boldsymbol{t}_s))$, such that for any $i \in [s]$, with $M_i = M_{i,1} \| \cdots \| M_{i,\ell}$ and $\boldsymbol{t}_i = t_{i,1} \| \cdots \| t_{i,\ell}$, for some $\ell \in \mathbb{N}$, (1) $|M'_i| = |M_i| = \ell n$, and (2) for any $j \in [\ell]$ such that $t_{i,j} = 0$, $M'_{i,j} = M_{i,j}$.

The intuition behind an R-LPSE scheme is that for each ciphertext $C = C_1 \| \cdots \| C_\ell$, for some $\ell \in \mathbb{N}$, tagged decryption will tag each deciphered block M_i, for $i \in [\ell]$, that is deemed ambiguous by setting t_i to 1 (and to 0 otherwise), so that the output is $(M_1 \| \cdots \| M_\ell, t_1 \| \cdots \| t_\ell)$. A block is deemed ambiguous if it signals a repetition, but no previous plaintext block can be found in the decryption state. After a batch of ciphertexts has been transmitted (that is, after a session has terminated), if the communication channel did not guarantee that the order was preserved, running the recovery algorithm on the batch can then resolve any such ambiguity.

2.4 Semantic Security

For some $n \in \mathbb{N}$, let Π be an LPSE scheme with key space $\mathcal{K}$ and message and ciphertext spaces $(\{0,1\}^n)^+$. We define semantic security of Π as *indistinguishability from random ciphertexts* (IND-CPA) (as introduced in [AR00, RBBK01]). Considering games $\mathrm{G}_{\Pi}^{\text{ind-cpa-0}}$ and $\mathrm{G}_{\Pi}^{\text{ind-cpa-1}}$ in Figure 1, we define the advantage of an IND-CPA adversary A as

$$\mathbf{Adv}_{\Pi}^{\text{ind-cpa}}(A) \doteq \Pr[\mathrm{G}_{\Pi}^{\text{ind-cpa-0}}(A)] - \Pr[\mathrm{G}_{\Pi}^{\text{ind-cpa-1}}(A)].$$

We let $\beta(A)$ denote the total number of n-bit blocks queried to ENC by A.[1]

2.5 Correctness

For some $n \in \mathbb{N}$, let Π be a (R-)LPSE scheme with key space $\mathcal{K}$ and message and ciphertext spaces $(\{0,1\}^n)^+$. We define two separate notions for correctness. The first models the setting where ciphertexts are *not* reordered in transit, and therefore if Π is just an LPSE (and not R-LPSE) scheme, it is the only correctness notion that can be achieved (if Π is an R-LPSE, it can also satisfy this notion). The second models the setting where ciphertexts *are* reordered in transit, and therefore it only applies in case that Π is an R-LPSE scheme.

2.5.1 Without Reordering of Ciphertexts

We define *correctness* (COR) (without reordering) of Π as the problem of distinguishing between an oracle that, given a message, returns the decryption of its encryption, and an

[1] This is a different measure than $\mu(A)$ from, e.g., [BDJR97], which measures how many *bits* were queried by A. We chose this measure for convenience, but the two are simply related by $\beta(A) = \mu(A)/n$.

Game $\mathrm{G}_{\Pi}^{\text{cor-0}}$	Game $\mathrm{G}_{\Pi}^{\text{cor-1}}$
procedure INIT	**procedure** ENCDEC(M)
$K \xleftarrow{\$} \mathcal{K}$	**return** M
$\mathbf{S}, \mathbf{T} \leftarrow [\,]$	
procedure ENCDEC(M)	
$C \leftarrow \Pi.\mathcal{E}_K^{\mathbf{S}}(M)$	
$M' \leftarrow \Pi.\mathcal{D}_K^{\mathbf{T}}(C)$	
return M'	

Figure 2: Games defining correctness *without* reordering of an LPSE scheme Π.

Game $\mathrm{G}_{\Pi}^{\text{cor-wr-0}}$

procedure INIT
  $K \xleftarrow{\$} \mathcal{K}$
  $\mathbf{S}, \mathbf{T} \leftarrow [\,]$
procedure ENCDECREC($M_1, \ldots, M_s, \pi$) // $\pi : [s] \to [s]$ must be a permutation.
  for $i = 1, \ldots, s$ **do**
    $C_i \leftarrow \Pi.\mathcal{E}_K^{\mathbf{S}}(M_i)$
  for $i = 1, \ldots, s$ **do**
    $(M'_{\pi(i)}, \boldsymbol{t}_{\pi(i)}) \leftarrow \Pi.\widetilde{\mathcal{D}}_K^{\mathbf{T}}(C_{\pi(i)})$
  $M''_{\pi(1)}, \ldots, M''_{\pi(s)} \leftarrow \mathcal{R}_K((M'_{\pi(1)}, \boldsymbol{t}_{\pi(1)}), \ldots, (M'_{\pi(s)}, \boldsymbol{t}_{\pi(s)}))$
  return $M''_{\pi(1)}, \ldots, M''_{\pi(s)}$

Game $\mathrm{G}_{\Pi}^{\text{cor-wr-1}}$

procedure ENCDECREC($M_1, \ldots, M_s, \pi$) // $\pi : [s] \to [s]$ must be a permutation.
  return $M_{\pi(1)}, \ldots, M_{\pi(s)}$

Figure 3: Games defining correctness *with* reordering of an R-LPSE scheme Π.

oracle that simply returns the queried message. Considering games $\mathrm{G}_{\Pi}^{\text{cor-0}}$ and $\mathrm{G}_{\Pi}^{\text{cor-1}}$ in Figure 2, we define the advantage of a COR adversary A as

$$\mathbf{Adv}_{\Pi}^{\text{cor}}(A) \doteq \Pr[\mathrm{G}_{\Pi}^{\text{cor-0}}(A)] - \Pr[\mathrm{G}_{\Pi}^{\text{cor-1}}(A)].$$

We let $\beta(A)$ denote the total number of n-bit blocks queried to ENCDEC by A.

2.5.2 With Reordering of Ciphertexts

Assume that Π is an R-LPSE. We define *correctness with reordering* (COR-WR) of Π as the problem of distinguishing between an oracle that, given a sequence of $s \in \mathbb{N}$ messages and a permutation on $[s]$, (1) encrypts the messages in the given order, (2) tag-decrypts them in the *permuted* order, (3) applies the recovery algorithm to the list of decrypted messages, and (4) returns the sequence of (permuted) recovered messages, and an oracle that simply returns the permuted sequence of queried message. Considering games $\mathrm{G}_{\Pi}^{\text{cor-wr-0}}$ and $\mathrm{G}_{\Pi}^{\text{cor-wr-1}}$ in Figure 3, we define the advantage of a COR-WR adversary A as

$$\mathbf{Adv}_{\Pi}^{\text{cor-wr}}(A) \doteq \Pr[\mathrm{G}_{\Pi}^{\text{cor-wr-0}}(A)] - \Pr[\mathrm{G}_{\Pi}^{\text{cor-wr-1}}(A)].$$

We let $\beta(A)$ denote the total number of n-bit blocks queried to ENCDECREC by A.

2.6 PRP Security

Let $\kappa, n \in \mathbb{N}$, $E : \{0,1\}^\kappa \times \{0,1\}^n \to \{0,1\}^n$, and for any $K \in \{0,1\}^\kappa$ and $M \in \{0,1\}^n$, define $E_K(M) \doteq E(K, M)$. Then E is a *block cipher* if, for any $K \in \{0,1\}^\kappa$, E_K is bijective, that is, E_K is a *permutation* on $\{0,1\}^n$. We say that E is a *pseudorandom permutation* (PRP) if it is indistinguishable from a uniformly selected permutation. We define the advantage of a PRP adversary A as

$$\mathbf{Adv}^{\mathrm{prp}}_E(A) \doteq \Pr[b = 0 \,|\, b \leftarrow A^{E_K}, K \xleftarrow{\$} \{0,1\}^\kappa] - \Pr[b = 0 \,|\, b \leftarrow A^{\pi}, \pi \xleftarrow{\$} \mathcal{P}_n].$$

We let $q(A)$ denote the total number of queries to E_K made by A.

2.7 Collision Resistance

Let $m, n \in \mathbb{N}$ with $m > n$, and $H : \{0,1\}^m \to \{0,1\}^n$. We say that H is a *collision resistant* (CR) *compression (hash) function* if it is hard to find two pre-images of H with the same image. We define the advantage of a CR adversary A as

$$\mathbf{Adv}^{\mathrm{cr}}_H(A) \doteq \Pr[X \neq Y \wedge H(X) = H(Y) \,|\, (X, Y) \leftarrow A].$$

Note that we consider unkeyed compression functions, which means that there always exists an efficient CR adversary with advantage 1. But as pointed out in [Rog06], this does not imply that one can actually *find* such adversary. Rather, in our proofs we give explicit constructions of CR adversaries from other adversaries (by means of a reduction). Still, it is possible to make our results more rigorous by letting compression functions be keyed. In this case, the target security would be *weak* collision resistance (WCR) from [BCK96].

3 Secure Codebook (SCB) Mode of Encryption

3.1 The Scheme

We design a stateful symmetric encryption scheme starting from a block cipher E and a compression function H, that is, we specify a *mode of encryption*. The main idea is to encipher each newly seen block normally with E_{K_1}, for some key K_1, and keep track of how many times each blocks has been seen so far via a look-up table $\mathbf{S}$ that maps blocks to integer counters; then, for each block that has been seen previously, rather than enciphering the block again (which is what ECB would do), we generate a hash value with H, append it to the bit string representing the number of times such block has been previously seen (retrieved from $\mathbf{S}$), pad with enough zeros, XOR with a second key K_2 of the same length as a block, and encipher the resulting bit string.

For decryption, we will also keep state by using a look-up table $\mathbf{T}$ mapping hash values to blocks. For each block of a ciphertext, we initially decipher it with $E^{-1}_{K_1}$, and then decide whether we believe the result to be the intended bit string upon encryption, in which case we store it under its hash value defined by H in $\mathbf{T}$, or whether it was a signal of a repetition. We always guess the latter case if the deciphered block has the right structure, that is, if it is appropriately zero-padded, and if the last bits correspond to a hash value that is contained in $\mathbf{T}$; In this case, simply retrieve the block from $\mathbf{T}$.

Clearly, things can go wrong, but we will show that under appropriate conditions, our scheme is still practical, and achieves semantic security. The first case in which correctness is violated, is if two blocks are mapped to the same hash value by H. Such a collision would force encryption to signal a wrong repetition, and the probability of such an event is upper-bounded by the collision resistance of H. The second case is if a block of a message to be encrypted is such that when XORed with K_2 has the structure of a repetition signal, and we will bound this event with a concrete probability.

$\mathsf{SCB}[E,H].\mathcal{E}^{\mathbf{S}}_{K_1,K_2}(M_1\|\cdots\|M_\ell)$	$\mathsf{SCB}[E,H].\mathcal{D}^{\mathbf{T}}_{K_1,K_2}(C_1\|\cdots\|C_\ell)$
1 : **for** $i=1,\ldots,\ell$ **do**	1 : **for** $i=1,\ldots,\ell$ **do**
2 : $\quad h \leftarrow H(M_i)$	2 : $\quad M_i \leftarrow E^{-1}_{K_1}(C_i)$
3 : $\quad$ **if** $\mathbf{S}[h]=\perp$ **then**	3 : $\quad R \leftarrow K_2 \oplus M_i$
4 : $\quad\quad C_i \leftarrow E_{K_1}(M_i)$	4 : $\quad h \leftarrow R \bmod 2^\tau$
5 : $\quad\quad \mathbf{S}[h] \leftarrow 0^\sigma$	5 : $\quad$ **if** $R < 2^{\sigma+\tau} \wedge \mathbf{T}[h] \neq \perp$ **then**
6 : $\quad$ **else**	6 : $\quad\quad M_i \leftarrow \mathbf{T}[h]$
7 : $\quad\quad R \leftarrow 0^{n-\sigma-\tau}\|\mathbf{S}[h]\|h$	7 : $\quad$ **else**
8 : $\quad\quad C_i \leftarrow E_{K_1}(K_2 \oplus R)$	8 : $\quad\quad h \leftarrow H(M_i)$
9 : $\quad\quad \mathbf{S}[h] \leftarrow (\mathbf{S}[h]+1) \bmod 2^\sigma$	9 : $\quad\quad \mathbf{T}[h] \leftarrow M_i$
10 : **return** $C_1\|\cdots\|C_\ell$	10 : **return** $M_1\|\cdots\|M_\ell$

Figure 4: Encryption and decryption algorithms of $\mathsf{SCB}[E,H]$.

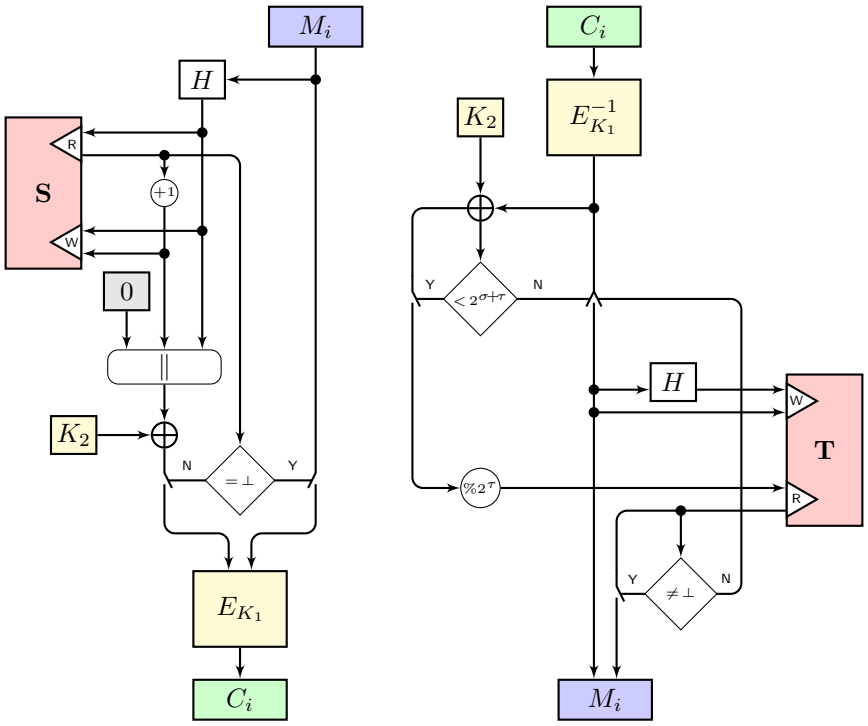


Figure 5: Schematic description of how encryption and decryption algorithms of $\mathsf{SCB}[E,H]$ process each block (we assume that $\perp + 1 \doteq 0$).

We now formally describe the scheme, and will prove its security and correctness in the next sections. We also provide a schematic description in Figure 5.

Definition 3. Let $\kappa, n, \sigma, \tau \in \mathbb{N}$ with $\sigma + \tau < n$, $E : \{0,1\}^\kappa \times \{0,1\}^n \to \{0,1\}^n$ a block cipher, and $H : \{0,1\}^n \to \{0,1\}^\tau$ a compression function. Also let $\mathcal{S}$ be the set of $\{0,1\}^\tau \to \{0,1\}^\sigma \cup \{\perp\}$ look-up tables, and $\mathcal{T}$ the set of $\{0,1\}^\tau \to \{0,1\}^n \cup \{\perp\}$ look-up tables. The *Secure Codebook* (SCB) mode of encryption is the LPSE scheme $\mathsf{SCB}[E,H] \doteq (\mathcal{E},\mathcal{D})$ with key space $\mathcal{K} = \{0,1\}^\kappa \times \{0,1\}^n$, encryption states space $\mathcal{S}$, decryption states space $\mathcal{T}$, and encryption and decryption algorithms $\mathcal{E}$ and $\mathcal{D}$ as defined in Figure 4.

Game G_0

1 : **procedure** INIT
2 : $K_1 \xleftarrow{\$} \{0,1\}^\kappa$
3 : $K_2 \xleftarrow{\$} \{0,1\}^n$
4 : $\mathbf{S} \leftarrow [\,]$
5 : **procedure** ENC(M)
6 : $C \leftarrow \mathsf{SCB}[E,H].\mathcal{E}^{\mathbf{S}}_{K_1,K_2}(M)$
7 : **return** C

Game G_3

1 : **procedure** ENC($M_1 \| \cdots \| M_\ell$)
2 : $C \xleftarrow{\$} \{0,1\}^{\ell n}$
3 : **return** C

Adversary B^{ENC}

1 : $K_2 \xleftarrow{\$} \{0,1\}^n$
2 : $\mathbf{S} \leftarrow [\,]$
3 : $b \leftarrow A^{\mathrm{ENC}^*}$
4 : **return** b
5 : **procedure** ENC*($M_1 \| \cdots \| M_\ell$)
6 : **for** $i = 1, \ldots, \ell$ **do**
7 : $h \leftarrow H(M_i)$
8 : **if** $\mathbf{S}[h] = \perp$ **then**
9 : $C_i \leftarrow \mathrm{ENC}(M_i)$
10 : $\mathbf{S}[h] \leftarrow 0^\sigma$
11 : **else**
12 : $R \leftarrow 0^{n-\sigma-\tau} \| \mathbf{S}[h] \| h$
13 : $C_i \leftarrow \mathrm{ENC}(K_2 \oplus R)$
14 : $\mathbf{S}[h] \leftarrow (\mathbf{S}[h] + 1) \bmod 2^\sigma$
15 : **return** $C_1 \| \cdots \| C_\ell$

Games G_1, $\boxed{\mathrm{G}_2}$

1 : **procedure** INIT
2 : $\pi \xleftarrow{\$} \mathcal{P}_n$
3 : $\boxed{\rho \xleftarrow{\$} \mathcal{F}_n}$
4 : $K_2 \xleftarrow{\$} \{0,1\}^n$
5 : $\mathbf{S} \leftarrow [\,]$
6 : $\mathbf{R}, \mathbf{M} \leftarrow [\,]$
7 : $\mathsf{bad} \leftarrow \mathsf{false}$
8 : $t \leftarrow 0$
9 : **procedure** ENC($M_1 \| \cdots \| M_\ell$)
10 : **for** $i = 1, \ldots, \ell$ **do**
11 : $h \leftarrow H(M_i)$
12 : **if** $\mathbf{S}[h] = \perp$ **then**
13 : **if** $\exists\, j < t :\ M_i = \mathbf{R}[j]$ **do**
14 : $\mathsf{bad} \leftarrow \mathsf{true}$
15 : $C_i \leftarrow \pi(M_i)$
16 : $\boxed{C_i \leftarrow \rho(M_i)}$
17 : $\mathbf{S}[h] \leftarrow 0^\sigma$
18 : $\mathbf{M}[t] \leftarrow K_2 \oplus M_i$
19 : **else**
20 : $R \leftarrow 0^{n-\sigma-\tau} \| \mathbf{S}[h] \| h$
21 : **if** $\exists\, j < t :\ R = \mathbf{M}[j]$ **do**
22 : $\mathsf{bad} \leftarrow \mathsf{true}$
23 : $C_i \leftarrow \pi(K_2 \oplus R)$
24 : $\boxed{C_i \leftarrow \rho(K_2 \oplus R)}$
25 : $\mathbf{S}[h] \leftarrow (\mathbf{S}[h] + 1) \bmod 2^\sigma$
26 : $\mathbf{R}[t] \leftarrow K_2 \oplus R$
27 : $t \leftarrow t + 1$
28 : **return** $C_1 \| \cdots \| C_\ell$

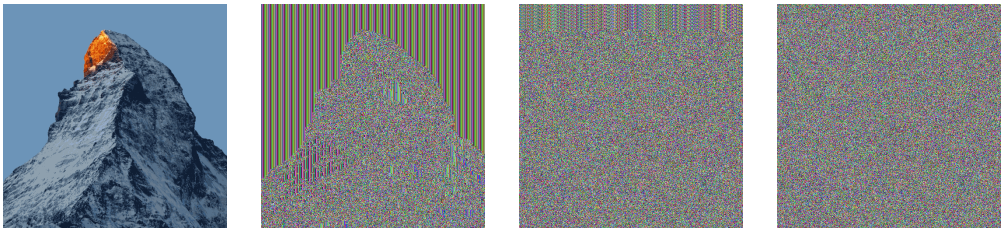
Figure 6: Games G_0–G_3 and adversary B for the proof of Theorem 1. Changes and additions from game G_0 to game G_1 are highlighted in the description of G_1, and the boxed code therein is exclusive to game G_2.

3.2 Security

Theorem 1. *For any* IND-CPA *adversary A with $\beta \doteq \beta(A) \leq 2^\sigma$ we can construct a* PRP *adversary B with $q(B) = \beta$ such that*

$$\mathbf{Adv}^{\text{ind-cpa}}_{\mathsf{SCB}[E,H]}(A) \leq \mathbf{Adv}^{\text{prp}}_{E}(B) + \frac{\beta^2}{2^n}.$$

Proof. Define games G_0–G_3 as in Figure 6. Note that, slightly abusing notation, we have $\mathrm{G}_0 = \mathrm{G}^{\text{ind-cpa-0}}_{\mathsf{SCB}[E,H]}$, $\mathrm{G}_1 = \mathrm{G}^{\text{ind-cpa-0}}_{\mathsf{SCB}[\mathcal{P}_n,H]}$, $\mathrm{G}_2 = \mathrm{G}^{\text{ind-cpa-0}}_{\mathsf{SCB}[\mathcal{F}_n,H]}$, and $\mathrm{G}_3 = \mathrm{G}^{\text{ind-cpa-1}}_{\mathsf{SCB}[E,H]}$. Moreover, G_2 and G_3 are identical until bad is set to true. To see this, observe that the two games differ in behavior only once ρ in G_2 is queried a certain value twice, since in that case the output of ENC is not an independent and uniformly random bit string. This can happen both at



(a) Original (b) ECB (c) $\sigma = 8, \tau = 32$ (d) $\sigma = 16, \tau = 32$

Figure 7: Visualization of the impact of the security parameter σ. A 512×512 image of Matterhorn (Figure 7a) has been encrypted with $K_1, K_2 = \texttt{thisisasecretkey}$[2] using $\mathsf{ECB}[\texttt{AES-128}]$ (Figure 7b) and $\mathsf{SCB}[\texttt{AES-128}, [\cdot]_\tau \circ \texttt{SHA-256}]$ for $\kappa = n = 128$, $\sigma = 8, 16$, and $\tau = 32$ (Figures 7c and 7d). The value chosen for τ guarantees perfect correctness. But clearly, both the ECB encrypted image and the SCB encrypted image with $\sigma = 8$ (note the repeating patterns in the top part) do not look pseudo-random, only the one encrypted with $\sigma = 16$ does. In fact, $\beta = 512 \times 512 \times 3 \div 16 = 49\,152 \leq 2^\sigma$ only for $\sigma = 16$.

lines 16 and 24, but crucially, any repeated query will be input to ρ once at line 16 and once at line 24. More precisely, since $\beta \leq 2^\sigma$, it is impossible to query ρ twice the same value at line 16 (this is guaranteed by the check at line 12), and it is also impossible to query ρ twice the same value at line 24 (this is guaranteed by the constantly increasing counter). Such collisions will happen with probability $\Pr[M_i = K_2 \oplus R] = 2^{-n}$, and multiplying by β, the number of times that ρ is queried in total, gives an upper bound on the probability that bad will be set to true. Furthermore, we have

$$\mathbf{Adv}^{\text{ind-cpa}}_{\mathsf{SCB}[E,H]}(A) = \Pr[\mathrm{G}_0(A)] - \Pr[\mathrm{G}_3(A)] = \sum_{i=0}^{2} (\Pr[\mathrm{G}_i(A)] - \Pr[\mathrm{G}_{i+1}(A)]).$$

Let adversary B be as in Figure 6. Then:

$$\begin{aligned}
\Pr[\mathrm{G}_0(A)] - \Pr[\mathrm{G}_1(A)] &= \Pr[\mathrm{G}_E^{\text{prp-0}}(B)] - \Pr[\mathrm{G}_E^{\text{prp-1}}(B)] \\
&= \mathbf{Adv}^{\text{prp}}_E(B), \\
\Pr[\mathrm{G}_1(A)] - \Pr[\mathrm{G}_2(A)] &= \Pr[B^\pi \mid \pi \xleftarrow{\$} \mathcal{P}_n] - \Pr[B^\rho \mid \rho \xleftarrow{\$} \mathcal{F}_n] \\
&\leq \frac{\beta(\beta-1)}{2^{n+1}}, \qquad (1) \\
\Pr[\mathrm{G}_2(A)] - \Pr[\mathrm{G}_3(A)] &\leq \Pr[\mathrm{G}_1(A) \text{ sets } \mathsf{bad}] \qquad (2) \\
&\leq \frac{\beta}{2^n},
\end{aligned}$$

where (1) follows by the PRP/PRF Switching Lemma [BR06, Lemma 1], and (2) follows by the Fundamental Lemma of Game Playing (FLGP) [BR06, Lemma 2]. Finally, since $\beta \geq 1$, we have

$$\frac{\beta(\beta-1)}{2^{n+1}} + \frac{\beta}{2^n} = \frac{\beta(\beta+1)}{2^{n+1}} \leq \frac{\beta^2}{2^n}. \qquad \square$$

In Figure 7 we provide a visual interpretation of Theorem 1. Note that the condition $\beta \leq 2^\sigma$ is a rough worst-case estimate. A more fine grained condition could depend on the transmitted messages, or better, on their distribution. For example, if it is known that among all transmitted messages, each block will not be repeated more than $N \leq 2^\sigma$ times, the condition would be unnecessary, and other more interesting properties of the distributions could significantly relax the original condition. We leave open the problem of improving the condition of Theorem 1 by taking into account the message distribution.

[2] We set $K_1 = K_2$ only for simplicity, this should *not* be done practice.

Game G_0

1 : **procedure** INIT
2 : $K_1 \xleftarrow{\$} \{0,1\}^\kappa$
3 : $K_2 \xleftarrow{\$} \{0,1\}^n$
4 : $\mathbf{S}, \mathbf{T} \leftarrow [\,]$
5 : **procedure** ENCDEC(M)
6 : $C \leftarrow \mathsf{SCB}[E,H].\mathcal{E}^{\mathbf{S}}_{K_1,K_2}(M)$
7 : $M' \leftarrow \mathsf{SCB}[E,H].\mathcal{D}^{\mathbf{T}}_{K_1,K_2}(C)$
8 : **return** M'

Game G_3

1 : **procedure** INIT
2 : $\mathbf{S} \leftarrow [\,]$
3 : **procedure** ENCDEC($M_1 \| \cdots \| M_\ell$)
4 : **for** $i = 1, \ldots, \ell$ **do**
5 : **if** $\mathbf{S}[M_i] = \bot$ **then**
6 : $C_i \leftarrow E_{K_1}(M_i)$
7 : $M'_i \leftarrow E^{-1}_{K_1}(C_i)$
8 : $\mathbf{S}[M_i] = \top$
9 : **else** // $\mathbf{S}[M_i] = \top$
10 : $C_i \leftarrow 0 \| M_i$
11 : $M'_i \leftarrow C_i \bmod 2^n$
12 : **return** $M'_1 \| \cdots \| M'_\ell$

Game G_4

1 : **procedure** ENCDEC(M)
2 : **return** M

Adversary B

1 : **procedure** INIT
2 : $X, Y \leftarrow \bot$
3 : $K_1 \xleftarrow{\$} \{0,1\}^\kappa$
4 : $K_2 \xleftarrow{\$} \{0,1\}^n$
5 : $\mathbf{S}, \mathbf{T}, \mathbf{M} \leftarrow [\,]$
6 : $t \leftarrow 0$
7 : **run** A^{ENCDEC^*}
8 : **return** (X, Y)
9 : **procedure** ENCDEC*($M_1 \| \cdots \| M_\ell$)
10 : **for** $i = 1, \ldots, \ell$ **do**
11 : $\mathbf{M}[t] \leftarrow M_i$
12 : $h \leftarrow H(M_i)$
13 : **if** $\exists j < t : M_i \neq \mathbf{M}[j]$
14 : $\wedge\ h = H(\mathbf{M}[j])$ **then**
15 : $(X, Y) \leftarrow (M_i, \mathbf{M}[j])$
16 : $t \leftarrow t + 1$
17 : $C \leftarrow \mathsf{SCB}[E,H].\mathcal{E}^{\mathbf{S}}_{K_1,K_2}(M_1 \| \cdots \| M_\ell)$
18 : $M' \leftarrow \mathsf{SCB}[E,H].\mathcal{D}^{\mathbf{T}}_{K_1,K_2}(C)$
19 : **return** M'

Figure 8: Games G_0, G_3, G_4 and adversary B for the proof of Theorem 2.

3.3 Correctness

Theorem 2. *For any* COR *adversary* A *with* $\beta \doteq \beta(A)$ *we can construct a* CR *adversary* B *with* $q(B) = \beta$ *such that*

$$\mathbf{Adv}^{\mathrm{cor}}_{\mathsf{SCB}[E,H]}(A) \leq \mathbf{Adv}^{\mathrm{cr}}_{H}(B) + \frac{2^\sigma \beta^2}{2^n}.$$

Proof. Define games G_0–G_4 as in Figures 8 and 9. Note that, $\mathrm{G}_0 = \mathrm{G}^{\text{cor-0}}_{\mathsf{SCB}[E,H]}$ and $\mathrm{G}_4 = \mathrm{G}^{\text{cor-1}}_{\mathsf{SCB}[E,H]}$. Moreover, G_0 and G_1 are equivalent, G_1 and G_2 are identical until bad_0 is set to true, G_2 and G_3 are identical until bad_1 is set to true, and G_3 and G_4 are equivalent. To see that G_1 and G_2 are identical until bad_0 is set to true, observe that the two games differ in behavior only once a collision in H is provoked, that is, a previous block ($\mathbf{M}[j]$) different than the current one ($M_i \neq \mathbf{M}[j]$) has the same hash value as the current one ($h = H(\mathbf{M}[j])$). As long as this event does not happen, no adversary can distinguish whether the game encodes and successively decodes repeated blocks (lines 19–20 and 24–28 in game G_1) or simply (internally) marks them by prepending a bit which it later ignores (lines 19 and 29 in game G_2). We will reduce provoking such collision to the collision resistance of H. To see that G_2 and G_3 are identical until bad_1 is set to true, observe that the two games differ in behavior only once a new block (that is, one that has not been queried before) has the structure of a repeated block, that is, when XORed with K_2 it has $n - \sigma - \tau$ leading zeros and the last τ bits correspond to the hash of a

Game G_1	Game G_2
procedure INIT	**procedure** INIT
$K_1 \xleftarrow{\$} \{0,1\}^\kappa$	$K_1 \xleftarrow{\$} \{0,1\}^\kappa$
$K_2 \xleftarrow{\$} \{0,1\}^n$	$K_2 \xleftarrow{\$} \{0,1\}^n$
$\mathbf{S}, \mathbf{T} \leftarrow [\,]$	$\mathbf{S}, \mathbf{T} \leftarrow [\,]$
$\mathbf{M} \leftarrow [\,]$	$\mathbf{M} \leftarrow [\,]$
$\mathsf{bad}_0 \leftarrow \mathsf{false}$	$\mathsf{bad}_1 \leftarrow \mathsf{false}$
$t \leftarrow 0$	$t \leftarrow 0$
procedure ENCDEC$(M_1 \| \cdots \| M_\ell)$	**procedure** ENCDEC$(M_1 \| \cdots \| M_\ell)$
for $i = 1, \ldots, \ell$ **do**	**for** $i = 1, \ldots, \ell$ **do**
$\mathbf{M}[t] \leftarrow M_i$	$\mathbf{M}[t] \leftarrow M_i$
$h \leftarrow H(M_i)$	**if** $\exists j < t, S \in \{0,1\}^\sigma : M_i =$
if $\exists j < t : M_i \neq \mathbf{M}[j]$	$K_2 \oplus 0^{n-\sigma-\tau} \| S \| H(\mathbf{M}[j])$
$\wedge\, h = H(\mathbf{M}[j])$ **then**	**then**
$\mathsf{bad}_0 \leftarrow \mathsf{true}$	$\mathsf{bad}_1 \leftarrow \mathsf{true}$
if $\mathbf{S}[h] = \bot$ **then**	**if** $\mathbf{S}[M_i] = \bot$ **then**
$C_i \leftarrow E_{K_1}(M_i)$	$C_i \leftarrow E_{K_1}(M_i)$
$\mathbf{S}[h] \leftarrow 0^\sigma$	$\mathbf{S}[M_i] = \top$
else	**else** // $\mathbf{S}[M_i] = \top$
$R \leftarrow 0^{n-\sigma-\tau} \| \mathbf{S}[h] \| h$	$C_i \leftarrow 0 \| M_i$
$C_i \leftarrow E_{K_1}(K_2 \oplus R)$	**for** $i = 1, \ldots, \ell$ **do**
$\mathbf{S}[h] \leftarrow (\mathbf{S}[h] + 1) \bmod 2^\sigma$	**if** $C_i \in \{0,1\}^n$ **then**
$t \leftarrow t + 1$	$M'_i \leftarrow E_{K_1}^{-1}(C_i)$
for $i = 1, \ldots, \ell$ **do**	$R \leftarrow K_2 \oplus M'_i$
$M'_i \leftarrow E_{K_1}^{-1}(C_i)$	$h \leftarrow R \bmod 2^\tau$
$R \leftarrow K_2 \oplus M'_i$	**if** $R < 2^{\sigma+\tau}$
$h \leftarrow R \bmod 2^\tau$	$\wedge\, \mathbf{T}[h] \neq \bot$ **then**
if $R < 2^{\sigma+\tau} \wedge \mathbf{T}[h] \neq \bot$ **then**	$M'_i \leftarrow \mathbf{T}[h]$
$M'_i \leftarrow \mathbf{T}[h]$	**else** // $C_i \in \{0,1\}^{1+n}$
else	$M'_i \leftarrow C_i \bmod 2^n$
$h \leftarrow H(M'_i)$	$h \leftarrow H(M'_i)$
$\mathbf{T}[h] \leftarrow M'_i$	$\mathbf{T}[h] \leftarrow M'_i$
return $M'_1 \| \cdots \| M'_\ell$	**return** $M'_1 \| \cdots \| M'_\ell$

Figure 9: Games G_1 and G_2 for the proof of Theorem 2. Changes and additions from game G_0 to game G_1 are highlighted in the description of G_1, and changes and additions from game G_1 to game G_2 are highlighted in the description of G_2.

previous block. By the union bound, we obtain that this happens with probability at most $\beta^2 \cdot 2^{-(n-\sigma-\tau)} \cdot 2^{-\tau} = \beta^2 \cdot 2^{\sigma-n}$. Furthermore, we have

$$\mathbf{Adv}^{\mathrm{cor}}_{\mathsf{SCB}[E,H]}(A) = \Pr[\mathrm{G}_0(A)] - \Pr[\mathrm{G}_4(A)] = \sum_{i=0}^{3} (\Pr[\mathrm{G}_i(A)] - \Pr[\mathrm{G}_{i+1}(A)]).$$

(a) Original (b) $\sigma = 16, \tau = 8$ (c) $\sigma = 16, \tau = 16$ (d) $\sigma = 16, \tau = 24$

Figure 10: Visualization of the impact of the correctness parameter τ. A 128×128 image of Tux (Figure 10a) has been encrypted and subsequently decrypted with $K_1, K_2 =$ `thisisasecretkey` using $\mathsf{SCB}[\texttt{AES-128}, [\cdot]_\tau \circ \texttt{SHA-256}]$ for $\kappa = n = 128$, $\sigma = 16$, and $\tau = 8, 16, 24$ (Figures 10b, 10c and 10d). The value chosen for σ guarantees that the condition of Theorem 1 is satisfied. For $\tau = 8$, too many collisions happened, so only certain patterns are visible; for $\tau = 16$, just a few collisions happened (six), but the decrypted image still has some errors; for $\tau = 24$ the original image was successfully recovered without any errors.

Let adversary B be as in Figure 8. Then:

$$\begin{aligned}
\Pr[\mathrm{G}_0(A)] - \Pr[\mathrm{G}_1(A)] &= 0, \\
\Pr[\mathrm{G}_1(A)] - \Pr[\mathrm{G}_2(A)] &\leq \Pr[\mathrm{G}_1(A) \text{ sets } \mathsf{bad}_0] \qquad (3) \\
&\leq \mathbf{Adv}^{\mathrm{cr}}_H(B), \\
\Pr[\mathrm{G}_2(A)] - \Pr[\mathrm{G}_3(A)] &\leq \Pr[\mathrm{G}_2(A) \text{ sets } \mathsf{bad}_1] \qquad (4) \\
&\leq \frac{2^\sigma \beta^2}{2^n}, \\
\Pr[\mathrm{G}_3(A)] - \Pr[\mathrm{G}_4(A)] &= 0,
\end{aligned}$$

where (3) and (4) follow by the FLGP. □

In Figure 10 we provide a visual interpretation of Theorem 2. Note that, even though the factor 2^σ in the second term of the correctness bound is undesirable, it still gives a meaningful result if σ is significantly smaller than n, and moreover can in principle be easily replaced by a better term. To see this, notice that we obtain the term because upon decryption, SCB simply ignores the counter, in case of a repetition. It should not be too hard to extend decryption in a way that counters are also accounted for, and in case reordering of ciphertexts might happen, some clever counter prediction technique should be implemented. Still, if ciphertexts do not get reordered, the term should in principle disappear. We leave open the problem of optimizing SCB even further and improve the factor 2^σ.

3.4 Extending SCB into a Variable-Input-Length LPSE Scheme

We now show how SCB can be easily extended into the first variable-input-length (VIL) length-preserving *encryption* (rather than enciphering) scheme achieving semantic security, denoted VIL-SCB. The idea is straightforward, we just need to apply the *ciphertext stealing* (CTS) paradigm (see e.g. [RWZ12]) to SCB. Informally, given a message $M \in \{0,1\}^{\geq n}$ of arbitrary length (but at least n), divide it into $\ell - 1$ blocks $M_1, \ldots, M_{\ell-1}$ of size n (defined by the block cipher) plus a last block M_ℓ of size $m \leq n$. If $m = n$, use regular SCB for all blocks, otherwise only encipher all but the last two blocks to obtain $C_1, \ldots, C_{\ell-2}$; then encipher the penultimate block $M_{\ell-1}$, and split the output into a bit string of length m and one of length $n - m$; the first m bits will form C_ℓ, whereas the last $n - m$ bits will be

VIL-SCB$[E,H].\mathcal{E}^{\mathbf{S}}_{K_1,K_2}(M)$

1 : $t \leftarrow |M|$
2 : $\ell \leftarrow \lceil t/n \rceil$
3 : $m \leftarrow t \bmod n$ // $m < n$
4 : $(M_1,\ldots,M_\ell) \leftarrow \mathsf{blocks}(M,n)$ // $M_1,\ldots,M_{\ell-1} \in \{0,1\}^n$, $M_\ell \in \{0,1\}^m$
5 : **for** $i = 1,\ldots,\ell-1$ **do**
6 : $C_i \leftarrow \mathsf{SCB}[E,H].\mathsf{Enc}^{\mathbf{S}}_{K_1,K_2}(M_i)$
7 : **if** $m = 0$ **do**
8 : $C_\ell \leftarrow \mathsf{SCB}[E,H].\mathsf{Enc}^{\mathbf{S}}_{K_1,K_2}(M_\ell)$
9 : **else**
10 : $(C_\ell, M_{\ell+1}) \leftarrow \mathsf{split}(C_{\ell-1}, m)$ // $C_\ell \in \{0,1\}^m$, $M_{\ell+1} \in \{0,1\}^{n-m}$
11 : $C_{\ell-1} \leftarrow \mathsf{SCB}[E,H].\mathsf{Enc}^{\mathbf{S}}_{K_1,K_2}(M_\ell \| M_{\ell+1})$ // $M_\ell \| M_{\ell+1} \in \{0,1\}^n$
12 : **return** $C_1 \| \cdots \| C_\ell$

VIL-SCB$[E,H].\mathcal{D}^{\mathbf{T}}_{K_1,K_2}(C)$

1 : $t \leftarrow |C|$
2 : $\ell \leftarrow \lceil t/n \rceil$
3 : $m \leftarrow t \bmod n$ // $m < n$
4 : $(C_1,\ldots,C_\ell) \leftarrow \mathsf{blocks}(C,n)$ // $C_1,\ldots,C_{\ell-1} \in \{0,1\}^n$, $C_\ell \in \{0,1\}^m$
5 : **for** $i = 1,\ldots,\ell-1$ **do**
6 : $M_i \leftarrow \mathsf{SCB}[E,H].\mathsf{Dec}^{\mathbf{T}}_{K_1,K_2}(C_i)$
7 : **if** $m = 0$ **do**
8 : $M_\ell \leftarrow \mathsf{SCB}[E,H].\mathsf{Dec}^{\mathbf{T}}_{K_1,K_2}(C_\ell)$
9 : **else**
10 : $(M_\ell, C_{\ell+1}) \leftarrow \mathsf{split}(M_{\ell-1}, m)$ // $M_\ell \in \{0,1\}^m$, $C_{\ell+1} \in \{0,1\}^{n-m}$
11 : $M_{\ell-1} \leftarrow \mathsf{SCB}[E,H].\mathsf{Dec}^{\mathbf{T}}_{K_1,K_2}(C_\ell \| C_{\ell+1})$ // $C_\ell \| C_{\ell+1} \in \{0,1\}^n$
12 : **return** $M_1 \| \cdots \| M_\ell$

Figure 11: Encryption and decryption algorithms of VIL-SCB$[E,H]$.

appended to M_ℓ, thus obtaining a bitstring of length n which will be enciphered to obtain $C_{\ell-1}$. Decryption works in the obvious way.

In order to formally describe VIL-SCB, let define $\mathsf{SCB}[E,H].\mathsf{Enc}^{\mathbf{S}}_{K_1,K_2}(M_i)$ as the code from lines 2–9 of $\mathsf{SCB}[E,H].\mathcal{E}^{\mathbf{S}}_{K_1,K_2}$ from Figure 4 plus the statement **return** C_i and $\mathsf{SCB}[E,H].\mathsf{Dec}^{\mathbf{T}}_{K_1,K_2}(C_i)$ as the code from lines 2–9 of $\mathsf{SCB}[E,H].\mathcal{D}^{\mathbf{T}}_{K_1,K_2}$ from Figure 4 plus the statement **return** M_i. Moreover, for a bit string S and an integer $i \le |S|$, let define $(S_1,\ldots,S_\ell) \leftarrow \mathsf{blocks}(S,i)$ such that $S_1\|\cdots\|S_\ell = S$, $S_1,\ldots,S_{\ell-1} \in \{0,1\}^i$, $S_\ell \in \{0,1\}^l$ if $l \doteq |S| \bmod n > 0$ and $S_\ell \in \{0,1\}^i$ otherwise, as well as $(S_1,S_2) \leftarrow \mathsf{split}(S,i)$ such that $S_1\|S_2 = S$, $S_1 \in \{0,1\}^i$, and $S_2 \in \{0,1\}^{|S|-i}$.

Formally, we consider VIL-SCB to be a VIL-LPSE scheme, which is just like a LPSE scheme from Definition 1, but with message and ciphertext spaces $\{0,1\}^{\ge n}$ rather than just $(\{0,1\}^n)^+$. Also, note that, referring to e.g. [Dwo10, RWZ12], we are using the CS2 type of CTS, that is, the one where the last two blocks are swapped if and only if the input message has length *not* multiple of n.

Definition 4. Let $\kappa, n, \sigma, \tau \in \mathbb{N}$ with $\sigma + \tau < n$, $E : \{0,1\}^\kappa \times \{0,1\}^n \to \{0,1\}^n$ a block cipher, and $H : \{0,1\}^n \to \{0,1\}^\tau$ a compression function. Also let $\mathcal{S}$ be the set of

$\{0,1\}^\tau \to \{0,1\}^\sigma \cup \{\bot\}$ look-up tables, and $\mathcal{T}$ the set of $\{0,1\}^\tau \to \{0,1\}^n \cup \{\bot\}$ look-up tables. The *Variable-Input-Length Secure Codebook* (VIL-SCB) mode of encryption is the VIL-LPSE scheme $\mathsf{VIL\text{-}SCB}[E,H] \doteq (\mathcal{E},\mathcal{D})$ with key space $\mathcal{K} = \{0,1\}^\kappa \times \{0,1\}^n$, encryption states space $\mathcal{S}$, decryption states space $\mathcal{T}$, and encryption and decryption algorithms $\mathcal{E}$ and $\mathcal{D}$ as defined in Figure 11.

Finally, note that if VIL-SCB is made stateless, that is, **S** and **T** are reset to the empty look-up table [] upon each encryption and decryption, respectively, then we obtain a length-preserving *enciphering* scheme (still with imperfect correctness), because now the scheme will only be computationally indistinguishable from uniform permutations (for each length at least n).

4 Recoverable SCB (RSCB) Mode of Encryption

4.1 The Scheme

We now describe how to extend SCB mode into an R-LPSE scheme, RSCB, by equipping it with two additional algorithms, *tagged decryption* and *recovery*. For the former, the idea is to enhance regular decryption by including a bit string of length corresponding to the number of blocks in the message/ciphertext, where each bit signals whether the corresponding block was ambiguous, that is, whether when XORed with K_2 it is zero-padded like a repetition signal. To recover a set of tagged messages, we then essentially run through all blocks twice; in the first pass, we reconstruct the look-up table **T**, and in the second pass we use **T** to check whether any ambiguous block had the structure of a repetition, but was received before the original block.

We now formally describe the scheme, and will prove its correctness *with reordering* in the next section. Note that we do not need to prove security, because it is inherited by Theorem 1 for SCB mode.

Definition 5. Let $\kappa, n, \sigma, \tau \in \mathbb{N}$ with $\sigma + \tau < n$, $E : \{0,1\}^\kappa \times \{0,1\}^n \to \{0,1\}^n$ a block cipher, and $H : \{0,1\}^n \to \{0,1\}^\tau$ a compression function. Also let $\mathcal{S}$ be the set of $\{0,1\}^\tau \to \{0,1\}^\sigma \cup \{\bot\}$ look-up tables, and $\mathcal{T}$ the set of $\{0,1\}^\tau \to \{0,1\}^n \cup \{\bot\}$ look-up tables. The *Recoverable Secure Codebook* (RSCB) mode of encryption is the R-LPSE scheme $\mathsf{RSCB}[E,H] \doteq (\mathcal{E},\mathcal{D},\widetilde{\mathcal{D}},\mathcal{R})$ with key space $\mathcal{K} = \{0,1\}^\kappa \times \{0,1\}^n$, encryption states space $\mathcal{S}$, decryption states space $\mathcal{T}$, encryption and decryption algorithms $\mathcal{E}$ and $\mathcal{D}$ as defined in Figure 4, and tagged decryption and recovery algorithms as defined in Figure 12.

We next provide a simple example to better understand how RSCB works.

Example 1. Let $M_1, \ldots, M_7 \in \{0,1\}^n$ be single-block messages, and define $h_i \doteq H(M_i)$ for $i \in [7]$. Moreover, assume that $M_4 = M_1$ (message M_4 is a repetition), $M_5 = K_2 \oplus h_2$ (M_5 collides with a true repetition signal), M_6 is such that $h_6 = h_3$ (M_6 causes a collision in H), and $M_7 = K_2 \oplus h$ with $h \neq h_i$ for all $i \in [7]$ (M_7 is a false repetition signal). Except for $M_4 = M_1$, assume all messages are different, and except for $h_4 = h_1$ and $h_6 = h_3$, also assume no other collisions in H happen. Also assume that $M_4 \neq K_2 \oplus h_1$ and $M_6 \neq K_2 \oplus h_3$. Now, encrypting the messages with $\mathcal{E}$ in the specified order, we obtain the following sequence of ciphertexts: $C_1 = E_{K_1}(M_1)$, $C_2 = E_{K_1}(M_2)$, $C_3 = E_{K_1}(M_3)$, $C_4 = E_{K_1}(K_2 \oplus h_1)$, $C_5 = E_{K_1}(K_2 \oplus h_2)$, $C_6 = E_{K_1}(K_2 \oplus h_3)$, $C_7 = E_{K_1}(K_2 \oplus h)$. Assuming that the seven ciphertexts are sent in the same order, we would obtain the following sequence of messages after decrypting them with $\mathcal{D}$: $M'_1 = M_1$, $M'_2 = M_2$, $M'_3 = M_3$, $M'_4 = M'_1 = M_1 = M_4$, $M'_5 = M'_2 = M_2 \neq M_5$, $M'_6 = M'_3 = M_3 \neq M_6$, $M'_7 = K_2 \oplus h = M_7$. Therefore, only the fifth and sixth messages would be compromised. We will next see that even if we permute the order in which we (tag-)decrypt the six ciphertexts, after applying the recovery algorithm $\mathcal{R}$ we would end up in the same state.

$\mathsf{RSCB}[E,H].\widetilde{\mathcal{D}}^{\mathbf{T}}_{K_1,K_2}(C_1\|\cdots\|C_\ell)$

1 : $M_1\|\cdots\|M_\ell \leftarrow \mathsf{RSCB}[E,H].\mathcal{D}^{\mathbf{T}}_{K_1,K_2}(C_1\|\cdots\|C_\ell)$
2 : **for** $i=1,\ldots,\ell$ **do**
3 : $\quad R \leftarrow K_2 \oplus M_i$
4 : $\quad$ **if** $R < 2^{\sigma+\tau}$ **then**
5 : $\quad\quad t_i \leftarrow 1$
6 : $\quad$ **else**
7 : $\quad\quad t_i \leftarrow 0$
8 : **return** $(M_1\|\cdots\|M_\ell, t_1\|\cdots\|t_\ell)$

$\mathsf{RSCB}[E,H].\mathcal{R}_{K_2}((M_{1,1}\|\cdots\|M_{1,\ell_1}, t_{1,1}\|\cdots\|t_{1,\ell_1}),\ldots,(M_{s,1}\|\cdots\|M_{s,\ell_s}, t_{s,1}\|\cdots\|t_{s,\ell_s}))$

1 : $\mathbf{T} \leftarrow [\,]$
2 : **for** $i=1,\ldots,s$ **do**
3 : $\quad$ **for** $j=1,\ldots,\ell_i$ **do**
4 : $\quad\quad h \leftarrow H(M_{i,j})$
5 : $\quad\quad \mathbf{T}[h] \leftarrow M_{i,j}$
6 : **for** $i=1,\ldots,s$ **do**
7 : $\quad$ **for** $j=1,\ldots,\ell_i$ **do**
8 : $\quad\quad h \leftarrow (K_2 \oplus M_{i,j}) \bmod 2^\tau$
9 : $\quad\quad$ **if** $t_{i,j}=1 \wedge \mathbf{T}[h] \neq \perp$ **then**
10 : $\quad\quad\quad M'_{i,j} \leftarrow \mathbf{T}[h]$
11 : $\quad\quad$ **else**
12 : $\quad\quad\quad M'_{i,j} \leftarrow M_{i,j}$
13 : **return** $(M'_{1,1}\|\cdots\|M'_{1,\ell_1},\ldots,M'_{s,1}\|\cdots\|M'_{s,\ell_s})$

Figure 12: Tagged decryption and recovery algorithms of $\mathsf{RSCB}[E,H]$. Note that $\mathcal{R}$ does not use K_1, so we slightly abused notation in the function declaration.

Now assume that the ciphertexts are tag-decrypted with $\widetilde{\mathcal{D}}$ in the following order instead: $C_4, C_5, C_6, C_7, C_1, C_2, C_3$. Then, we would obtain the following sequence of message-bit pairs: $(M'_4, 1)$ with $M'_4 = K_2 \oplus h_1 \neq M_4$, $(M'_5, 1)$ with $M'_5 = K_2 \oplus h_2 = M_5$, $(M'_6, 1)$ with $M'_6 = K_2 \oplus h_3 \neq M_6$, $(M'_7, 1)$ with $M'_7 = K_2 \oplus h = M_7$, $(M'_1, 0)$ with $M'_1 = M_1$, $(M'_2, 0)$ with $M'_2 = M_2$, $(M'_3, 1)$ with $M'_3 = M_3$. Therefore, in this case M_5 would not have been compromised, but M_4 and M_6 would. Let now $h'_i \doteq H(M'_i)$, for $i \in [7]$. When applying the recovery algorithm $\mathcal{R}$, we will first build the table $\mathbf{T}$ such that $\mathbf{T}[h'_i] = M'_i$, for $i \in [7]$. Hence, we have that $\mathbf{T}[(K_2 \oplus M'_4) \bmod 2^\tau] = \mathbf{T}[h_1] = \mathbf{T}[h'_1] = M'_1 = M_1$, $\mathbf{T}[(K_2 \oplus M'_5) \bmod 2^\tau] = \mathbf{T}[h_2] = \mathbf{T}[h'_2] = M'_2 = M_2$, $\mathbf{T}[(K_2 \oplus M'_6) \bmod 2^\tau] = \mathbf{T}[h_3] = \mathbf{T}[h'_3] = M'_3 = M_3$, $\mathbf{T}[(K_2 \oplus M'_7) \bmod 2^\tau] = \mathbf{T}[h] = \perp$. Therefore, after the second pass in $\mathcal{R}$, the following sequence of messages will be output: $M''_4 = M_1$, $M''_5 = M_2$, $M''_6 = M_3$, $M''_7 = M_7$, $M''_1 = M_1$, $M''_2 = M_2$, $M''_3 = M_3$. Indeed, each individual message was now decrypted the same way it would have been decrypted if the transmitted ciphertexts were not permuted. ◊

Note that for the specific case of RSCB, tags are actually not really necessary since they can be also computed directly by $\mathcal{R}$, but in practice it might still be beneficial to the overall efficiency of the scheme to compute them directly upon decryption. Finally, note that RSCB can also be easily transformed into a VIL-LPSE scheme, VIL-RSCB, simply by following the same approach used for SCB as outlined in Section 3.4.

4.2 Correctness (with Reordering)

Theorem 3. *For any* COR-WR *adversary* A *with* $\beta \doteq \beta(A)$ *we can construct a* CR *adversary* B *with* $q(B) = \beta$ *such that*

$$\mathbf{Adv}^{\text{cor-wr}}_{\mathsf{SCB}[E,H]}(A) \leq \mathbf{Adv}^{\text{cr}}_{H}(B) + \frac{2^\sigma \beta^2}{2^n}.$$

Proof (sketch). We only provide a high level idea of the proof, since for the most part it can be reduced to that of Theorem 2. More specifically, one only needs to show that it is possible to remove line 9 and replace line 8 with the statement $M_i'' \leftarrow \mathsf{SCB}[E,H].\mathcal{D}^{\mathbf{T}}_{K_1,K_2}(C_i)$ in game $\mathrm{G}^{\text{cor-wr-0}}_{\mathsf{SCB}[E,H]}$. After this modification, it is easy to see that we can further remove lines 7–8 and replace line 6 with the statement $M_i'' \leftarrow \mathrm{G}^{\text{cor-0}}_{\mathsf{SCB}[E,H]}.\textsc{EncDec}(M_i)$ as well as lines 2–3 with the statement $\mathrm{G}^{\text{cor-wr-0}}_{\mathsf{SCB}[E,H]}.\textsc{Init}$. This naturally induces a reduction to the problem of distinguishing between $\mathrm{G}^{\text{cor-0}}_{\mathsf{SCB}[E,H]}$ and $\mathrm{G}^{\text{cor-1}}_{\mathsf{SCB}[E,H]}$, from which we get the bound, since once we finally replace line 6 with the statement $M_i'' \leftarrow \mathrm{G}^{\text{cor-1}}_{\mathsf{SCB}[E,H]}.\textsc{EncDec}(M_i)$ (which guarantees $M_i'' = M_i$), we obtain a game that behaves identically to $\mathrm{G}^{\text{cor-wr-1}}_{\mathsf{SCB}[E,H]}$.

To see that we can indeed apply the first modification to $\mathrm{G}^{\text{cor-wr-0}}_{\mathsf{SCB}[E,H]}$ without changing the behavior of the game, observe that for each block we have five cases: (1) the block is new, (2) the block is a repetition, (3) the block has the structure of a repetition signal for another transmitted block, (4) the block is new but its hash value collides with the hash of a previous block, and (5) the block has the structure of a repetition signal but for a block that was not transmitted. Then, as we saw in Example 1, it is easy to see that given a sequence of messages and a permutation, if we apply $\mathcal{E}$ to the sequence and then permute the resulting ciphertexts, for each of the five cases the deciphered block is the same whether we apply $\widetilde{\mathcal{D}}$ and then $\mathcal{R}$ to the permuted ciphertext sequence, or just $\mathcal{D}$. □

5 Practical Considerations

5.1 Security-Correctness Trade-Off

In the following we assume that SCB is instantiated with a *keyed* compression function $H : \{0,1\}^\lambda \times \{0,1\}^n \to \{0,1\}^\tau$, for some $\lambda \in \mathbb{N}$. Hence we set the key space to be $\mathcal{K} = \{0,1\}^\kappa \times \{0,1\}^n \times \{0,1\}^\lambda$, and adapt encryption and decryption algorithms to use key $(K_1, K_2, K_3) \in \mathcal{K}$ in the obvious way, that is, simply by replacing calls to H by calls to H_{K_3}. Referring to our discussion in Section 2.7, this means that H must satisfy *weak* collision resistance (WCR) [BCK96, Definition 3.1], which means we should simply replace $\mathbf{Adv}^{\text{cr}}_H(B)$ by $\mathbf{Adv}^{\text{wcr}}_H(B')$ in Theorem 2, for an appropriate WCR adversary B'. The reason for assuming a keyed compression function, is that in order for our analysis below to be sound, we need adversaries *not* being able to perform offline queries to H. Still, as we will discuss later, our notion of correctness is quite strong, and to us it does not look so problematic if in practice H is instantiated without keys.

Towards finding a trade-off between security and correctness, first note that both bounds from Theorems 1 and 2 do not (explicitly) depend on the correctness parameter τ, but exclusively on the block length n, the security parameter σ, and the total number of transmitted blocks β. Clearly, the dominating factor is $\frac{2^\sigma \beta^2}{2^n}$ from Theorem 2, so we should *minimize* σ. Since from Theorem 1 we have the condition $\beta \leq 2^\sigma$, we can derive lower and upper bounds on σ, given n and β:

$$\log \beta \leq \sigma \ll n - 2 \log \beta.$$

Because of the birthday bound (BB), an implicit condition on Theorem 2 is that $\beta \leq 2^{\frac{\tau}{2}}$. But the BB also allows us to roughly lower bound the term $\mathbf{Adv}^{\text{wcr}}_H(B')$ by $\beta^2/2^\tau$. Therefore,

since this implies that we should *maximize* τ, and since $\sigma + \tau < n$, we can derive lower and upper bounds on τ as well, given n, σ, and β:

$$2 \log \beta \ll \tau \leq n - \sigma.$$

Note that setting $\tau = n - \sigma$ would imply that the condition $R < 2^{\sigma+\tau}$ at line 5 of $\mathsf{SCB}[E, H].\mathcal{D}^{\mathbf{T}}_{K_1,K_2}$ from Figure 4 would always be true, as $n = \sigma + \tau$, but for efficiency reasons, it might be still better *not* to set $\tau = n - \sigma$, since this would imply less look-ups in $\mathbf{T}$ on average.

For concreteness, suppose that we have $n = 128$ and that we estimate that the total number of transmitted blocks will not exceed $\beta = 2^{10}$. Then, a reasonable choice of parameters would be $\sigma = 10$ and $\tau = 108$, since in this case $2^{\sigma}\beta^2/2^n = \beta^2/2^{\tau} = 2^{-98}$. This essentially gives us 97 bits of security, and since we set τ to be strictly less than $n - \sigma$, we also avoid to perform a look-up in $\mathbf{T}$ for every block upon decryption. But if we would instead estimate $\beta = 2^{20}$, then the best we can do is to set $\sigma = 20$ and $\tau = n - \sigma = 108$, since in this case $2^{\sigma}\beta^2/2^n = \beta^2/2^{\tau} = 2^{-68}$. This essentially gives us 67 bits of security, but we have to perform a look-up in $\mathbf{T}$ for every block upon decryption.

Finally, we remark that in principle it is possible to arbitrarily increase the level of security by simply first applying any conventional length-preserving enciphering technique to E in order to increase its block length, and then instantiating SCB with the resulting block cipher. Clearly, this would be at the cost of overall increased time complexity.

5.2 Possible Modifications, Extensions, and Optimizations

An obvious modification to SCB could be to specify the key space as $\mathcal{K} \doteq \{0,1\}^{\kappa}$, and then use a pseudorandom generator $\mathsf{PRG} : \{0,1\}^{\kappa} \to \{0,1\}^{\kappa+n}$ to obtain $K_1 \| K_2 \leftarrow \mathsf{PRG}(K)$ from $K \in \mathcal{K}$ upon initialization. This would be more in line with SCB being a wrapper around ECB, since the key space of the latter is just $\mathcal{K}$ from the block cipher E. Clearly, this would naturally extend to the case where the compression function H is keyed by some key $K_3 \in \{0,1\}^{\lambda}$ simply by using a pseudorandom generator $\mathsf{PRG}' : \{0,1\}^{\kappa} \to \{0,1\}^{\kappa+n+\lambda}$ instead.

A possible extension to SCB could be to compute $E_{K_1}(K_2 \oplus R) \oplus K_2$ instead of $E_{K_1}(K_2 \oplus R)$ at line 8 of $\mathsf{SCB}[E, H].\mathcal{E}^{\mathbf{S}}_{K_1,K_2}$ from Figure 4, following the approach of the Even-Mansour scheme [EM93, DKS12]. This could be a first step towards extending SCB into a CCA secure scheme. Moreover, as discussed Section 3.3, it should not be too hard to extend SCB decryption in a way that counters are also accounted for, and in case reordering of ciphertexts might happen, some clever counter prediction technique could allow to correctly guess that some blocks signal repetitions, even if they are transmitted later than the original block.

In order to gain some efficiency, line 8 of $\mathsf{SCB}[E, H].\mathcal{E}^{\mathbf{S}}_{K_1,K_2}$ could be replaced by $E_{K_2}(R)$ instead (and $\mathsf{SCB}[E, H].\mathcal{D}^{\mathbf{T}}_{K_1,K_2}$ correspondingly adapted). The downside of this approach, is that the resulting scheme would not be a wrapper around ECB anymore. Finally, to construct the compression function H efficiently one could for example follow the approach of [SS08], and then use H itself to implement the look-up tables $\mathbf{S}$ and $\mathbf{T}$.

5.3 Further Remarks

In our notion of correctness, we only considered the case where encryption and decryption use the same security and correctness parameters σ and τ. Interestingly, there should enough room for σ and τ such that encrypting and decrypting with (slightly) mismatching parameters, should yield a satisfactory level of correctness. Moreover, if parties anyway cannot pre-agree on σ and τ, the sender could simply set a good enough σ and try increasing the value of τ and decrypt itself the ciphertext until it sees no errors, and the receiver can then do the same, until increasing τ does not change the decrypted message anymore.

Note that our notion of correctness is very strong: We assume that a sender could send a potentially bad message, that is, one containing a block M such that $K_2 \oplus M < 2^{\sigma+\tau}$. Clearly, a better bound than the one from Theorem 2 could be derived, if we simply impose that all such blocks are forbidden from being part of sent messages. But we feel that this is too strong of a requirement in practice, considering that anyway we also still perform the check $\mathbf{T}[h] \neq \perp$ at line 5 of $\mathsf{SCB}[E, H].\mathcal{D}^{\mathbf{T}}_{K_1,K_2}$ from Figure 4, which on average would fail most of the times.

We remark that SCB is in principle parallelizable, if thread-safe look-up tables are available. The only caveat is that each thread should process all its blocks twice, to make sure that repetitions of blocks processed by other threads are correctly accounted for. Moreover, a further advantage of SCB is that error propagation is limited to single blocks, unlike other modes such as CBC where two blocks are affected.

Finally, we believe that our scheme has efficiency comparable to that of most VIL (tweakable) LPE schemes from the literature, while achieving better security. The code of $\mathsf{SCB}[\texttt{AES-128}, [\cdot]_\tau \circ \texttt{SHA-256}]$ used to generate the images of Figures 7 and 10 is available on https://github.com/fbanfi90/scb. A concrete estimation of the efficiency of this particular instantiation of SCB could be carried out using the results from [KS06].

6 Conclusions and Future Work

In this paper we initiated the study of the subtle trade-off between security and correctness of symmetric encryption, if it is required that ciphertexts preserve the length of plaintexts. To achieve this goal, we focused on the Electronic Codebook (ECB) mode of encryption, and enhanced it into a mode, Secure Codebook (SCB), that does not have perfect correctness, but achieves semantic security. Even if it only has imperfect correctness, SCB is still practical because its parameters can be set such that the probability of errors upon decryption is negligible. Nevertheless, as previously remarked, both security and correctness bounds we provided for SCB could in principle be improved by slight modifications of the scheme and analysis, and we leave this task open for future work.

Clearly, there might be other ways to achieve semantically secure length-preserving encryption, and we also leave open the question of finding other schemes with potentially better security-correctness trade-offs. One particular aspect of SCB that might limit its practicality is that the state linearly grows on each subsequent encryption and decryption. Therefore a natural question that arises is whether it is possible to design a semantically secure length-preserving encryption with *constant size state*. We suspect that this is not possible (for reasonable sizes), or that it would require further trade-offs. Nevertheless, it might be the case that schemes with better state growth rate than SCB exist.

We analyzed two different settings for correctness: without and with reordering of ciphertexts. A further notion of correctness that could be introduced and studied for SCB, and in general for future LPSE schemes, is one that considers *deletion of ciphertexts* as well. For SCB, we would clearly still have a significant advantage over the CTR approach outlined in the introduction, with respect to such a notion. Other interesting future directions might be for example enhancing SCB in a way that it achieves CCA security or even extend it into a (nonce-based) authenticated encryption scheme. For the latter, there might be a connection to the VIL tweakable LPE schemes from the literature.

Acknowledgments

The author would like to thank Ueli Maurer, Mihir Bellare, and Giovanni Deligios for the helpful discussions and feedback, as well as the anonymous ToSC reviewers for their insightful comments that helped improve the quality of the paper.

References

[AR00] Martín Abadi and Phillip Rogaway. Reconciling two views of cryptography. In Jan van Leeuwen, Osamu Watanabe, Masami Hagiya, Peter D. Mosses, and Takayasu Ito, editors, *TCS 2000*, volume 8895 of *LNCS*, pages 3–22. Springer, Heidelberg, August 2000. doi:10.1007/3-540-44929-9_1.

[BCK96] Mihir Bellare, Ran Canetti, and Hugo Krawczyk. Keying hash functions for message authentication. In Neal Koblitz, editor, *CRYPTO'96*, volume 1109 of *LNCS*, pages 1–15. Springer, Heidelberg, August 1996. doi:10.1007/3-540-68697-5_1.

[BD99] Daniel Bleichenbacher and Anand Desai. A construction of a super-pseudorandom cipher. *Manuscript, February*, 1999.

[BDJR97] Mihir Bellare, Anand Desai, Eric Jokipii, and Phillip Rogaway. A concrete security treatment of symmetric encryption. In *38th FOCS*, pages 394–403. IEEE Computer Society Press, October 1997. doi:10.1109/SFCS.1997.646128.

[BKR94] Mihir Bellare, Joe Kilian, and Phillip Rogaway. The security of cipher block chaining. In Yvo Desmedt, editor, *CRYPTO'94*, volume 839 of *LNCS*, pages 341–358. Springer, Heidelberg, August 1994. doi:10.1007/3-540-48658-5_32.

[BR99] Mihir Bellare and Phillip Rogaway. On the construction of variable-input-length ciphers. In Lars R. Knudsen, editor, *FSE'99*, volume 1636 of *LNCS*, pages 231–244. Springer, Heidelberg, March 1999. doi:10.1007/3-540-48519-8_17.

[BR06] Mihir Bellare and Phillip Rogaway. The security of triple encryption and a framework for code-based game-playing proofs. In Serge Vaudenay, editor, *EUROCRYPT 2006*, volume 4004 of *LNCS*, pages 409–426. Springer, Heidelberg, May / June 2006. doi:10.1007/11761679_25.

[CB18] Paul Crowley and Eric Biggers. Adiantum: length-preserving encryption for entry-level processors. *IACR Trans. Symm. Cryptol.*, 2018(4):39–61, 2018. doi:10.13154/tosc.v2018.i4.39-61.

[CKY07] Debra L. Cook, Angelos D. Keromytis, and Moti Yung. Elastic block ciphers: the basic design. In Feng Bao and Steven Miller, editors, *ASIACCS 07*, pages 350–352. ACM Press, March 2007.

[CN08] Debrup Chakraborty and Mridul Nandi. An improved security bound for HCTR. In Kaisa Nyberg, editor, *FSE 2008*, volume 5086 of *LNCS*, pages 289–302. Springer, Heidelberg, February 2008. doi:10.1007/978-3-540-71039-4_18.

[CS06] Debrup Chakraborty and Palash Sarkar. A new mode of encryption providing a tweakable strong pseudo-random permutation. In Matthew J. B. Robshaw, editor, *FSE 2006*, volume 4047 of *LNCS*, pages 293–309. Springer, Heidelberg, March 2006. doi:10.1007/11799313_19.

[CS08] Debrup Chakraborty and Palash Sarkar. HCH: A new tweakable enciphering scheme using the hash-counter-hash approach. *IEEE Transactions on Information Theory*, 54(4):1683–1699, 2008. doi:10.1109/TIT.2008.917623.

[CYK04] Debra L. Cook, Moti Yung, and Angelos D. Keromytis. Elastic block ciphers. Cryptology ePrint Archive, Report 2004/128, 2004. https://eprint.iacr.org/2004/128.

[DKS12] Orr Dunkelman, Nathan Keller, and Adi Shamir. Minimalism in cryptography: The Even-Mansour scheme revisited. In David Pointcheval and Thomas Johansson, editors, *EUROCRYPT 2012*, volume 7237 of *LNCS*, pages 336–354. Springer, Heidelberg, April 2012. doi:10.1007/978-3-642-29011-4_21.

[Dwo10] Morris Dworkin. Recommendation for block cipher modes of operation: Three variants of ciphertext stealing for CBC mode, 2010-10-21 2010. URL: https://tsapps.nist.gov/publication/get_pdf.cfm?pub_id=906929.

[EM93] Shimon Even and Yishay Mansour. A construction of a cipher from a single pseudorandom permutation. In Hideki Imai, Ronald L. Rivest, and Tsutomu Matsumoto, editors, *ASIACRYPT'91*, volume 739 of *LNCS*, pages 210–224. Springer, Heidelberg, November 1993. doi:10.1007/3-540-57332-1_17.

[GM82] Shafi Goldwasser and Silvio Micali. Probabilistic encryption and how to play mental poker keeping secret all partial information. In *14th ACM STOC*, pages 365–377. ACM Press, May 1982. doi:10.1145/800070.802212.

[Hal04] Shai Halevi. EME*: Extending EME to handle arbitrary-length messages with associated data. In Anne Canteaut and Kapalee Viswanathan, editors, *INDOCRYPT 2004*, volume 3348 of *LNCS*, pages 315–327. Springer, Heidelberg, December 2004.

[Hal07] Shai Halevi. Invertible universal hashing and the TET encryption mode. In Alfred Menezes, editor, *CRYPTO 2007*, volume 4622 of *LNCS*, pages 412–429. Springer, Heidelberg, August 2007. doi:10.1007/978-3-540-74143-5_23.

[HR03] Shai Halevi and Phillip Rogaway. A tweakable enciphering mode. In Dan Boneh, editor, *CRYPTO 2003*, volume 2729 of *LNCS*, pages 482–499. Springer, Heidelberg, August 2003. doi:10.1007/978-3-540-45146-4_28.

[HR04] Shai Halevi and Phillip Rogaway. A parallelizable enciphering mode. In Tatsuaki Okamoto, editor, *CT-RSA 2004*, volume 2964 of *LNCS*, pages 292–304. Springer, Heidelberg, February 2004. doi:10.1007/978-3-540-24660-2_23.

[KS06] Jens-Peter Kaps and Berk Sunar. Energy comparison of aes and sha-1 for ubiquitous computing. In Xiaobo Zhou, Oleg Sokolsky, Lu Yan, Eun-Sun Jung, Zili Shao, Yi Mu, Dong Chun Lee, Dae Young Kim, Young-Sik Jeong, and Cheng-Zhong Xu, editors, *EUC 2006*, volume 4097 of *LNCS*, pages 372–381. Springer, Heidelberg, August 2006. doi:10.1007/11807964_38.

[LRW02] Moses Liskov, Ronald L. Rivest, and David Wagner. Tweakable block ciphers. In Moti Yung, editor, *CRYPTO 2002*, volume 2442 of *LNCS*, pages 31–46. Springer, Heidelberg, August 2002. doi:10.1007/3-540-45708-9_3.

[MF04] David A. McGrew and Scott R. Fluhrer. The extended codebook (XCB) mode of operation. Cryptology ePrint Archive, Report 2004/278, 2004. https://eprint.iacr.org/2004/278.

[MF07] David A. McGrew and Scott R. Fluhrer. The security of the extended codebook (XCB) mode of operation. In Carlisle M. Adams, Ali Miri, and Michael J. Wiener, editors, *SAC 2007*, volume 4876 of *LNCS*, pages 311–327. Springer, Heidelberg, August 2007. doi:10.1007/978-3-540-77360-3_20.

[MM07] Kazuhiko Minematsu and Toshiyasu Matsushima. Tweakable enciphering schemes from hash-sum-expansion. In K. Srinathan, C. Pandu Rangan, and Moti Yung, editors, *INDOCRYPT 2007*, volume 4859 of *LNCS*, pages 252–267. Springer, Heidelberg, December 2007.

[Nan08] Mridul Nandi. Improving upon HCTR and matching attacks for hash-counter-hash approach. Cryptology ePrint Archive, Report 2008/090, 2008. https://eprint.iacr.org/2008/090.

[PRS04] Sarvar Patel, Zulfikar Ramzan, and Ganapathy S. Sundaram. Efficient constructions of variable-input-length block ciphers. In Helena Handschuh and Anwar Hasan, editors, *SAC 2004*, volume 3357 of *LNCS*, pages 326–340. Springer, Heidelberg, August 2004. doi:10.1007/978-3-540-30564-4_23.

[RBBK01] Phillip Rogaway, Mihir Bellare, John Black, and Ted Krovetz. OCB: A block-cipher mode of operation for efficient authenticated encryption. In Michael K. Reiter and Pierangela Samarati, editors, *ACM CCS 2001*, pages 196–205. ACM Press, November 2001. doi:10.1145/501983.502011.

[Rog04] Phillip Rogaway. Nonce-based symmetric encryption. In Bimal K. Roy and Willi Meier, editors, *FSE 2004*, volume 3017 of *LNCS*, pages 348–359. Springer, Heidelberg, February 2004. doi:10.1007/978-3-540-25937-4_22.

[Rog06] Phillip Rogaway. Formalizing human ignorance. In Phong Q. Nguyen, editor, *Progress in Cryptology - VIETCRYPT 06*, volume 4341 of *LNCS*, pages 211–228. Springer, Heidelberg, September 2006.

[RWZ12] Phillip Rogaway, Mark Wooding, and Haibin Zhang. The security of ciphertext stealing. In Anne Canteaut, editor, *FSE 2012*, volume 7549 of *LNCS*, pages 180–195. Springer, Heidelberg, March 2012. doi:10.1007/978-3-642-34047-5_11.

[Sar07] Palash Sarkar. Improving upon the TET mode of operation. In Kil-Hyun Nam and Gwangsoo Rhee, editors, *ICISC 07*, volume 4817 of *LNCS*, pages 180–192. Springer, Heidelberg, November 2007.

[SS08] Thomas Shrimpton and Martijn Stam. Building a collision-resistant compression function from non-compressing primitives. In Luca Aceto, Ivan Damgård, Leslie Ann Goldberg, Magnús M. Halldórsson, Anna Ingólfsdóttir, and Igor Walukiewicz, editors, *ICALP 2008, Part II*, volume 5126 of *LNCS*, pages 643–654. Springer, Heidelberg, July 2008. doi:10.1007/978-3-540-70583-3_52.

[ST13] Thomas Shrimpton and R. Seth Terashima. A modular framework for building variable-input-length tweakable ciphers. In Kazue Sako and Palash Sarkar, editors, *ASIACRYPT 2013, Part I*, volume 8269 of *LNCS*, pages 405–423. Springer, Heidelberg, December 2013. doi:10.1007/978-3-642-42033-7_21.

[WFW05] Peng Wang, Dengguo Feng, and Wenling Wu. HCTR: A variable-input-length enciphering mode. In Dengguo Feng, Dongdai Lin, and Moti Yung, editors, *CISC 2005*, volume 3822 of *LNCS*, pages 175–188. Springer, Heidelberg, 2005. doi:10.1007/11599548_15.