On Compression of Data Encrypted with Block Ciphers

Demijan Klinc, Carmit Hazay, Ashish Jagmohan, Hugo Krawczyk, and Tal Rabin

Abstract

This paper investigates compression of data encrypted with block ciphers, such as the Advanced Encryption Standard (AES). It is shown that such data can be feasibly compressed without knowledge of the secret key. Block ciphers operating in various chaining modes are considered and it is shown how compression can be achieved without compromising security of the encryption scheme. Further, it is shown that there exists a fundamental limitation to the practical compressibility of block ciphers when no chaining is used between blocks. Some performance results for practical code constructions used to compress binary sources are presented.

Index Terms

Compression, encrypted data, block ciphers, CBC mode, ECB mode, Slepian-Wolf coding.

I. Introduction

We consider the problem of compressing encrypted data. Traditionally in communication systems, data from a source is first compressed and then encrypted before it is transmitted over a channel to the receiver. While in many cases this approach is befitting, there exist scenarios where there is a need to reverse the order in which data encryption and compression are performed. Consider for instance a network of low-cost sensor nodes that transmit sensitive information over the internet to a recipient. The sensor nodes need to encrypt data to hide it from potential eavesdroppers, but they may not be able to perform compression as that would require additional hardware and thus higher implementation cost. On

D. Klinc is with the Georgia Institute of Technology, Atlanta, GA. Email: demi@ece.gatech.edu

C. Hazay is with Bar-Ilan University, Ramat-Gan, Israel. Email: harelc@cs.biu.ac.il

A. Jagmohan, H. Krawczyk, and T. Rabin are with IBM T.J. Watson Research Labs, Yorktown Heights and Hawthorne, NY. Email: {ashishja, talr}@us.ibm.com, hugo@ee.technion.ac.il

the other hand, the network operator that is responsible for transfer of data to the recipient wants to compress the data to maximize the utilization of its resources. It is important to note that the network operator is not trusted and hence does not have access to the key used for encryption and decryption. If it had the key, it could simply decrypt data, compress and encrypt again.

We focus on compression of encrypted data where the encryption procedure utilizes block ciphers such as the Advanced Encryption Standard (AES) [1] and Data Encryption Standard (DES) [2]. Loosely speaking, block ciphers operate on inputs of fixed length and serve as important building blocks that can be used to construct secure encryption schemes.

For a fixed key a block cipher is a bijection, therefore the entropy of an input is the same as that of the output. It follows that it is theoretically possible to compress the source to the same level as before encryption. However, in practice, encrypted data appears to be random and the conventional compression techniques do not yield desirable results. Consequently, it was long believed that encrypted data is practically incompressible. In a surprising paper [3], the authors break that paradigm and show that the problem of compressing one-time pad encrypted data translates to the problem of compressing correlated sources, which was solved by Slepian and Wolf in [4] and for which practical and efficient codes are known. Compression is practically achievable due to a simple symbol-wise correlation between the key (one-time pad) and the encrypted message. However, when such correlation is more complex, as is the case with block ciphers, the approach to Slepian-Wolf coding utilized in [3] is not directly applicable.

In this paper, we investigate if data encrypted with block ciphers can be compressed without access to the key. We show that block ciphers in conjunction with the most commonly used chaining modes in practice (e.g., [5], [6]) are practically compressible for some types of sources. To our knowledge this is the first work to show that non-negligible compression gains can be achieved for cryptographic algorithms like AES or DES when they are used in non-stream modes; in particular, this work offers a solution to the open problem formulated in [7, Sec. 3.3].

Moreover, by using standard techniques in the cryptographic literature we prove that the proposed compression schemes do not compromise the security of the original encryption scheme. We also show that there exists a fundamental limitation to the compression capability when the input to the block cipher is applied to a single-block message without chaining as in ECB mode (see Section III-B).

The outline of this paper is as follows. Section II defines the problem that we seek to solve and summarizes existing work on the subject. Section III focuses on block ciphers and explains how they can be compressed without knowledge of the secret key in various modes of operation. Section IV discusses security of the proposed compression scheme, while Section V touches on the subject of compressing data

encrypted with public-key encryption schemes. Section VI presents some simulation results for binary memoryless sources and finally, Section VII concludes the paper.

II. PRELIMINARIES

We begin with a standard formal definition of an encryption scheme as stated in [8]. A private-key encryption scheme is a triple of algorithms (Gen, Enc, Dec), where Gen is a probabilistic algorithm that outputs a key K chosen according to some distribution that is determined by the scheme; the encryption algorithm Enc takes as input a key K and a plaintext message X and outputs a ciphertext $Enc_K(X)$; the decryption algorithm Dec takes as input a key K and a ciphertext $Enc_K(X)$ and outputs a plaintext $Dec_K(Enc_K(X)) = X$. It is required that for every key K output by Gen and every plaintext X, we have $Dec_K(Enc_K(X)) = X$.

In private-key encryption schemes of concern to us in this paper the same key is used for encryption and decryption algorithms. Private-key encryption schemes can be divided in two categories: block ciphers and stream ciphers. Stream ciphers encrypt plaintext one symbol at a time, typically by summing it with a key (XOR operation for binary alphabets). In contrast, block ciphers represent a different approach where encryption is accomplished by means of nonlinear mappings on input blocks of fixed length. Common examples of block ciphers are AES and DES. Typically, block ciphers are not used as a stand-alone encryption procedure. Instead, they are combined to work on variable-length data using composition mechanisms known as chaining modes or modes of operation, as specified in Section III.

We proceed with a formulation of the source coding problem with decoder side-information, which is illustrated in Figure 1. Consider random variables X (termed the source), and S (termed the side-information), both over a finite-alphabet and with a joint probability distribution P_{XS}. Consider a sequence of independent n realizations of (X, S) denoted by $\{X_i, S_i\}_{i=1}^n$.

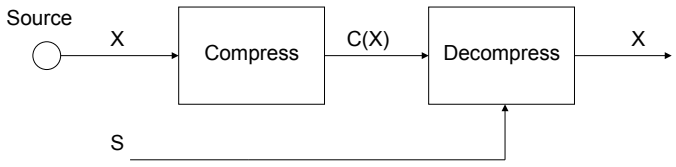

Fig. 1. Lossless source coding with decoder side-information.

The problem at hand is that of losslessly encoding $\{X_i\}_{i=1}^n$, with $\{S_i\}_{i=1}^n$ known only to the decoder. In [4], Slepian and Wolf showed that, for sufficiently large block length n, this can be done at rates arbitrarily

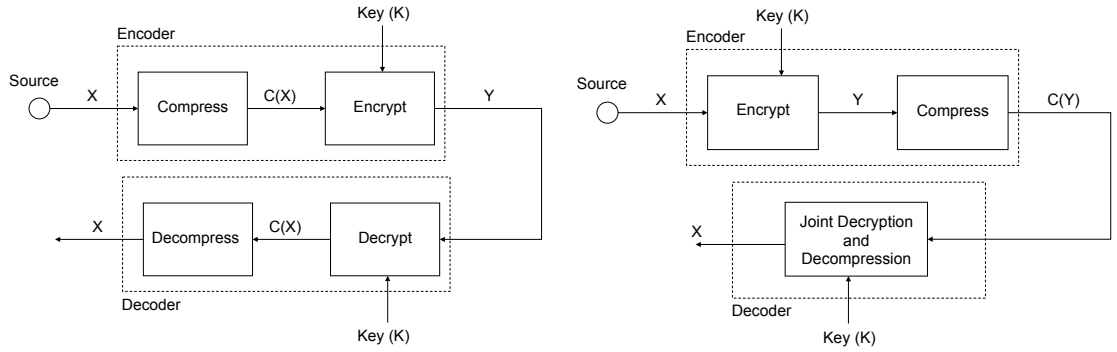

Fig. 2. Systems combining compression and encryption. (a) Traditional system with compression done first. (b) System with encryption done before compression.

close to the conditional entropy $H(X|S)$. Practical Slepian-Wolf coding schemes use constructions based on good linear error-correcting codes [9]–[11].

In this paper, we are interested in systems that perform both compression and encryption, wherein the compressor does not have access to the key. Typically, in such systems, encryption is performed after compression as depicted in Figure 2(a). This is a consequence of the traditional view which considers ciphertext data hard to compress without knowledge of the key. In [3] a system similar to Figure 2(b) is considered instead, in which the order of the encryption and compression operations at the encoder is reversed (note, though, that only the encryptor has access to the key). The authors consider encryption of a plaintext X using a one-time pad scheme, with a finite-alphabet key (pad) K, to generate the ciphertext Y, i.e.

$$Y_i \triangleq X_i \oplus K_i.$$

This is followed by compression, which is agnostic of K, to generate the compressed ciphertext $C(Y)$.

The key insight underlying the approach in [3] is that the problem of compression in this case can be formulated as a Slepian-Wolf coding problem. In this formulation the ciphertext Y is cast as a source, and the shared key K is cast as the decoder-only side-information. The joint distribution of the source and side-information can be determined from the statistics of the source. For example, in the binary case with a uniformly distributed K_i and X_i with $\Pr[X_i = 1] = p$, we have

$$\Pr(Y_i \neq k | K_i = k) = p. \tag{1}$$

The decoder has knowledge of K and of the source statistics. It uses this knowledge to reconstruct the ciphertext Y from the compressed message $C(Y)$, and to subsequently decrypt the plaintext X. This

formulation is leveraged in [3] to show that exactly the same lossless compression rate, $H(X)$, can be asymptotically achieved in the system shown in Figure 2(b), as can be achieved in Figure 2(a). Further, this can be done while maintaining information-theoretic security.

The one-time pad and stream ciphers, while convenient for analysis, are not the only forms of encryption in practice. In fact, the prevalent method of encryption uses block ciphers, thus an obviously desirable extension of the technique in [3] would be to conventional encryption schemes such as the popular AES encryption method. Attempting to do so, however, proves to be problematic. The method in [3] leverages the fact that in a one-time pad encryption scheme there exists a simple symbol-wise correlation between the key K and the ciphertext Y, as seen in (1). Unfortunately, for block ciphers such as AES no such correlation structure is known. Moreover, any change in the plaintext is diffused in the ciphertext, and quantifying the correlation (or the joint probability distribution) of the key and the ciphertext is believed to be computationally infeasible.

In the remainder of this paper, we show how this problem can be circumvented by exploiting the chaining modes popularly used with block ciphers. Based on this insight, we present an approach for compressing data encrypted with block ciphers, without knowledge of the key. As in [3], the proposed methods are based on the use of Slepian-Wolf coding.

Before continuing, we formalize the notion of a "post-encryption compression" which is used througout this paper with the following definition:

Definition 1 (Post-Encryption Compression (PEC) Scheme): Let $\mathcal{E} = (Gen, Enc, Dec)$ be an encryption scheme, as defined above, with plaintext domain $\mathcal{X}$ and ciphertext range $\mathcal{Q}$, and let $\mathcal{P}$ be a probability distribution over $\mathcal{X}$. Let C be a compression function defined over $\mathcal{Q}$, and D be a (possibly probabilistic) decoding function with the property that, for any key K generated by Gen, $D_K(C(Enc_K(X)) = X$ with probability $1 - \delta$. We call the pair (C, D) an $(\mathcal{E}, \mathcal{P}, \delta)$-PEC scheme.

Note that in this definition, D is given access to the encryption key while C works independently of the key. Both C and D may be built for a specific distribution $\mathcal{P}$ (in particular, correct decoding may be guaranteed, with high probability, only for plaintexts chosen according to a specific $\mathcal{P}$). We often assume that the plaintext distribution $\mathcal{P}$ is efficiently samplable, which means that there exists an efficient randomized algorithm whose output distribution is $\mathcal{P}$. The probability of error δ is taken over the choice of X and the choice of random coins if D is randomized. To simplify notation, we shall omit the $(\mathcal{E}, \mathcal{P}, \delta)$ parameters in the exposition if they are irrelevant or evident from the context, and use the term PEC scheme.

PEC schemes may be tailored to a specific cipher, say AES or DES. Most often, however, one is interested in schemes that can support different ciphers, for instance both AES and DES, or even a full family of encryption schemes. All PEC schemes presented in this paper can work with any block cipher, therefore we call them generic PEC schemes. A generic PEC scheme cannot be tailored to the specific details of the underlying cipher but rather handles the cipher as a black box. That is, it is not necessary that the decoder D knows the specifics of the encryption scheme or even its encryption/decryption key. Rather, it suffices that D has access to a pair of encryption and decryption oracles, denoted by Enc and Dec, that provide D with encryptions and decryptions, respectively, of any plaintext or ciphertext queried by D.

III. Compressing Block-cipher Encryption

In contrast to stream ciphers, such as the one-time pad, block ciphers are highly nonlinear and the correlation between the key and the ciphertext is, by design, hard to characterize. If a block cipher operates on each block of data individually, two identical inputs will produce two identical outputs. While this weakness does not necessarily enable an unauthorized user to understand contents of an individual block it can reveal valuable information; for example, about frequently occurring data patterns. To address this problem, various chaining modes, also called modes of operation, are used in conjunction with block ciphers. The idea is to randomize each plaintext block, by using a randomization vector derived as a function of previous encryptor inputs or outputs. The randomization prevents two identical plaintext blocks from being encrypted into two identical ciphertext blocks, thus preventing leakage of information about data patterns.

We are interested in the following problem. Consider a sequence of plaintext blocks $\mathbf{X}^n = \{X_i\}_{i=1}^n$, where each block X_i is drawn from the set $\mathcal{X}^m = \{0,1\}^m$. Further, we assume that the blocks X_i are generated by an i.i.d. source with a distribution P_X. The blocks in $\mathbf{X}^n$ are encrypted with a block-cipher based private-key encryption scheme (Gen, Enc, Dec). In most cases of interest, block-cipher based encryption schemes use an initialization vector IV that is drawn uniformly at random from $\mathcal{X}^m$ by the encryption algorithm Enc_K. Let the encryption algorithm be characterized by the mapping $Enc_K : (\mathcal{X}^m)^n \to \mathcal{X}^m \times (\mathcal{X}^m)^n$. For a sequence of plaintext blocks $\mathbf{X}^n$ at input, the encryption algorithm generates $Enc_K(\mathbf{X}^n) = \{\text{IV}, \mathbf{Y}^n\}$, where $\mathbf{Y}^n = \{Y_i\}_{i=1}^n$ denotes a sequence of ciphertext blocks and each block $Y_i \in \mathcal{X}^m$. The problem at hand is to compress $Enc_K(\mathbf{X}^n)$ without knowledge of K.

In the remainder of this section we focus on the cipher block chaining (CBC) mode and the electronic code book (ECB) mode. The CBC mode is interesting because it is the most common mode of operation

used with block ciphers (e.g. in Internet protocols TLS and IPsec), while our treatment of the ECB mode provides fundamental insight about the feasibility of performing compression on data compressed with block ciphers without chaining. Two other modes of operation associated with block ciphers are worth mentioning: the output feedback (OFB) mode and the cipher feedback (CFB) mode. The solution to compressing the latter two modes is a relatively straightforward extension of the methods from [3], therefore it is presented in Appendix A.

A. Cipher Block Chaining (CBC)

The most common mode of operation is CBC. Depicted in Figure 3, block ciphers in CBC mode are employed as the default mechanism in widespread security standards such as IPSec [5] and TLS/SSL [6] and hence it is a common method of encrypting internet traffic.

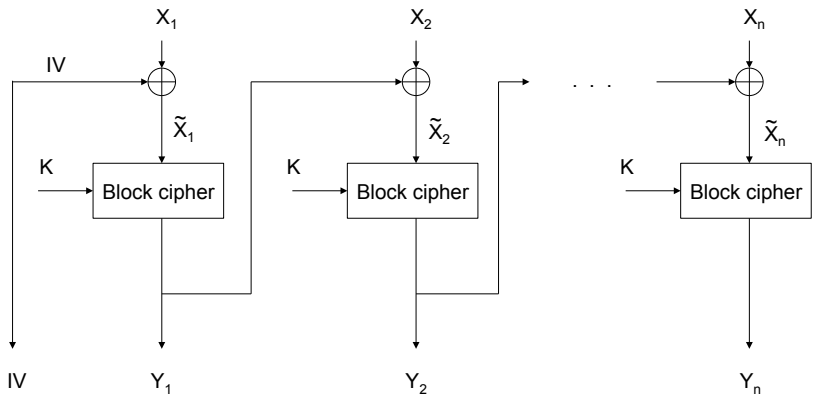

Fig. 3. Cipher block chaining (CBC).

In the CBC mode, each plaintext block X_i is randomized prior to encryption, by being XOR-ed with the ciphertext block corresponding to the previous plaintext block Y_{i-1}, to obtain $\tilde{X}_i$. Ciphertext block Y_i is generated by applying the block cipher with key K to the randomized plaintext block $\tilde{X}_i$

$$Y_i = B_K\big(X_i \oplus Y_{i-1}\big), \tag{2}$$

where $Y_0 = \text{IV}$ and $B_K : \mathcal{X}^m \to \mathcal{X}^m$ is the block cipher mapping using the key K. At the output of the encryption algorithm we have $Enc_K(\mathbf{X}^n) = (\text{IV}, \mathbf{Y}^n)$.

Notice that, contrary to modes OFB and CFB, ciphertext blocks in $\mathbf{Y}^n$ are not obtained by means of a bitwise XOR operation. Instead, they are obtained as outputs of highly nonlinear block ciphers, therefore the methods from [3] cannot be applied directly to compress in CBC mode.

The key insight underlying the proposed approach for compression can now be described. The statistical relationship between the key K and the i-th ciphertext block Y_i is hard to characterize. However, the joint distribution of the randomization vector Y_{i-1} and the i-th input to the block cipher, $\tilde{X}_i$, is easier to characterize, as it is governed by the distribution of the plaintext block X_i. For example, in the i.i.d source case we are considering, Y_{i-1} and $\tilde{X}_i$ are related through a symbol-wise model governed by the distribution P_X. The correlation induced by the use of the chaining mode can be exploited to allow compression of encrypted data using Slepian-Wolf coding, as we will now show.

Let $(C_{\text{CBC}}, D_{\text{CBC}})$ denote a Slepian-Wolf code with encoding rate R and block length m. The Slepian-Wolf encoding function is defined as $C_{\text{CBC}} : \mathcal{X}^m \rightarrow \{1, \ldots, 2^{mR}\}$, and the Slepian-Wolf decoding function as $D_{\text{CBC}} : \{1, \ldots, 2^{mR}\} \times \mathcal{X}^m \rightarrow \mathcal{X}^m$. The proposed compression method is illustrated in Figure 4. The input to the compressor is $Enc_K(\mathbf{X}^n) = (\text{IV}, \mathbf{Y}^n)$. Since $Y_i \in \mathcal{X}^m$, the total length of the input

Fig. 4. Compressor.

sequence is $(n+1) \cdot m \cdot \log |\mathcal{X}|$ bits. The compressor applies the Slepian-Wolf encoding function C_{CBC} to the IV and each of the first $n-1$ ciphertext blocks independently, while the n-th block is left unchanged. Thus, the output of the compressor is the sequence $(C_{\text{CBC}}(\text{IV}), C_{\text{CBC}}(Y_1), \ldots C_{\text{CBC}}(Y_{n-1}), Y_n)$. The length of the output sequence is $n \cdot m \cdot R + m \cdot \log |\mathcal{X}|$ bits. Thus, the compressor achieves a compression factor of

$$\lim_{n \rightarrow \infty} \frac{(n+1) \cdot m \cdot \log |\mathcal{X}|}{n \cdot m \cdot R + m \cdot \log |\mathcal{X}|} = \frac{\log |\mathcal{X}|}{R}$$

for large n. Note that the compressor does not need to know the key K. Also, note that this approach only requires a compressed IV, which by itself is incompressible, therefore no performance loss is inflicted by the uncompressed last block.

The joint decompression and decryption method is shown in Figure 5. The received compressed sequence is decrypted and decompressed serially, from right to left. In the first step Y_n, which is received uncompressed, is decrypted using the key K to generate $\tilde{X}_n$. Next, Slepian-Wolf decoding is performed to reconstruct Y_{n-1} using $\tilde{X}_n$ as side-information, and the compressed bits $C_{\text{CBC}}(Y_{n-1})$. The decoder computes $\hat{Y} \triangleq D_{\text{CBC}}(C_{\text{CBC}}(Y_{n-1}), \tilde{X}_n)$, such that $\hat{Y} = Y_{n-1}$ with high-probability if the rate R is high

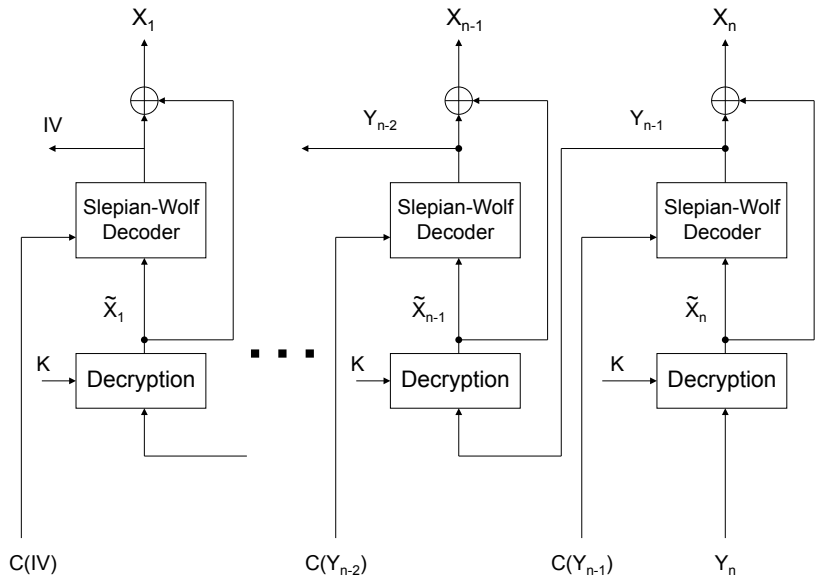

Fig. 5. Joint decryption and decoding at the receiver. It is performed serially from right to left.

enough. Once Y_{n-1} has been recovered by the Slepian-Wolf decoder, the plaintext block can now be reconstructed as $X_n = Y_{n-1} \oplus \tilde{X}_n$. The decoding process now proceeds serially with Y_{n-1} decrypted to generate $\tilde{X}_{n-1}$, which acts as the new Slepian-Wolf side-information. This continues until all plaintext blocks have been reconstructed.

For large m, it follows from the Slepian-Wolf theorem that the rate required to ensure correct reconstruction of the $(i-1)$-th block with high probability is given as

$$\begin{aligned} mR &= H\big(Y_{i-1}|\tilde{X}_i\big) = H\big(Y_{i-1}|Y_{i-1} \oplus X_i\big) \\ &= H\big(Y_{i-1}, Y_{i-1} \oplus X_i|Y_{i-1} \oplus X_i\big) = H\big(Y_{i-1}, X_i|Y_{i-1} \oplus X_i\big) \\ &= H\big(X_i|Y_{i-1} \oplus X_i\big) \leq H(X_i). \end{aligned} \tag{3}$$

It is assumed that the IV is drawn uniformly at random, therefore Y_i is uniformly distributed for all i. Consequently, the equation 3 holds with equality and we have $R = \frac{1}{m}H(X_i)$.

In practice, as we will see in Section 4, m is typically small. In this case, the required rate R is a function of P_X, m, the acceptable decoding error probability, and the non-ideal Slepian-Wolf codes used.

B. Electronic Code Book (ECB)

We have seen that ciphertexts generated by a block cipher in OFB, CFB, and CBC modes can be compressed without knowledge of the encryption key. The compression schemes that we presented rely on the specifics of chaining operations. A natural question at this point is to what extent can the output of a block cipher be compressed without chaining, i.e. when a block cipher is applied to a single block of plaintext. This mode of operation, depicted in Figure 6, is called the electronic code book (ECB) mode.

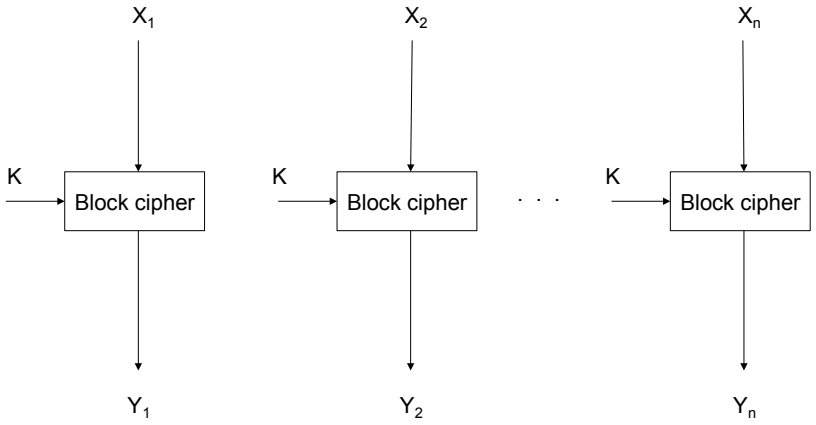

Fig. 6. Electronic Codebook (ECB).

Block cipher with a fixed key is a permutation, therefore the entropy of the ciphertext is the same as the entropy of the plaintext. In effect, compression in ECB mode is theoretically possible. The question is, whether it is possible to design a *generic and efficient* post-encryption compression scheme such as with other modes of operation.

We claim the answer to this question is negative, except for some low-entropy distributions or very low compression rates (e.g., compressing a ciphertext by a few bits). We show that for a given compressed ciphertext $C(Y_i)$, the decoder cannot do significantly better than to operate in one of the following two decoding strategies, which we refer to as *exhaustive* strategies:

1) enumerate all possible plaintexts in decreasing order of probability and compute a ciphertext for each plaintext until the ciphertext that compresses to $C(Y_i)$ is found;
2) enumerate all ciphertexts that compress to $C(Y_i)$ and decrypt them to find the original plaintext.

More precisely, we show that a generic compression scheme that compresses the output of a block cipher and departs significantly from one of the above strategies can be converted into an algorithm that breaks the security of the block cipher. In other words, a scheme that compresses the output of a secure block

cipher either requires an infeasible amount of computation, i.e. as much as needed to break the block cipher, or it must follow one of the above exhaustive strategies.

Definition 2: If (C, D) is a PEC scheme that follows any of the above exhaustive strategies, we say that (C, D) is an *exhaustive PEC scheme.*

Our result on the compressibility of block ciphers in ECB mode can now be stated.

Theorem 3: Let B be a secure block cipher[1], let $\mathcal{E} = (Gen, Enc, Dec)$ be an encryption scheme, where Gen chooses K uniformly from $\mathcal{K}$ and $Enc = B_K$, $Dec = B_K^{-1}$, and let $\mathcal{P}$ be an efficiently-samplable plaintext distribution. Let (C, D) be a $(\mathcal{E}, \mathcal{P}, \delta)$-PEC scheme. If (C, D) is generic for block ciphers, then (C, D) is either exhaustive or computationally infeasible (or both).

The exact computational bounds are omitted in the Theorem's statement for simplicity. More details on the computational bounds are given in Appendix C (Lemma 11), where Theorem 3 is also proven. The following remarks are worth noting:

(a) The exhaustive strategies are infeasible in most cases. However, for very low-entropy plaintext distributions or for very low compression rates (e.g., when compressing a ciphertext by just a few bits) these strategies can be efficient. For example, consider a plaintext distribution consisting of a set $\mathcal{X}$ of 1,000 128-bit values uniformly distributed. In this case, one can compress the output of a 128-bit block cipher applied to this set by truncating the 128-bit ciphertext to 40 bits. Using a birthday-type bound one can show that the probability of two values in $\mathcal{X}$ being mapped to the same ciphertext is about 2^{-20}. Hence, exhaustive strategy 1 above would succeed in recovering the correct plaintext with probability $1 - 2^{-20}$. In general, the compression capabilities under this strategy will depend on the *guessing entropy* [12], [13] of the underlying plaintext distribution. Another compression approach is to drop some of the bits in the ciphertext and let the decoder search exhaustively, as in the strategy 2 above, for the original ciphertext until the correct plaintext is recovered. This would assume that the number of dropped bits allows for almost-unique decodability and that the decoding procedure can identify the right plaintext. The fact that these exhaustive compression strategies may be efficient for some plaintext distributions shows that our theorem cannot exclude the existence of efficient generic coding schemes for some plaintext distributions. Rather, the theorem says that if there exists an efficient generic coding scheme for a given distribution, then it must follow one of the exhaustive strategies.

[1]For a definition of block cipher security please refer to Appendix B.

(b) Compressing the ciphertext is theoretically possible. If efficiency is not of concern, one could consider a brute-force compression algorithm that first breaks the block cipher by finding its key[2], uses the key to decrypt, and then compresses the plaintext. If several blocks of plaintext are compressed sequentially, they can be re-encrypted by the compression algorithm resulting in an effective ciphertext compression. This idea might be impractical, but it shows that if the existence of generic compressors is ruled out, this must be tied to the efficiency of the PEC scheme. Moreover, the above approach could work efficiently against an insecure block cipher (in which case one may find the key by efficient means), thus showing that *both efficiency and security* are essential ingredients to our result.

(c) Theorem 3 holds for generic compressors that do not use the internals of an encryption algorithm or the actual key in the (de)compression process. It does not rule out the existence of good PEC schemes for specific secure block ciphers, like for instance AES. To achieve compression, though, the PEC scheme would have to be contingent on the internal structure of a block cipher.

IV. Security of the CBC Compression Scheme

In this section we formally prove that compression and decompression operations that we introduced on top of the regular CBC mode do not compromise security of the original CBC encryption. The proof follows standard techniques in the cryptographic literature, showing that any efficient attack against secrecy of a PEC scheme can be transformed into an efficient attack against the original CBC encryption. Please refer to Appendix B for background on cryptographic definitions that are used in this section.

We start by formalizing the notion of security of a PEC scheme as a simple extension of the standard definition of chosen plaintext attack (CPA) security recalled in Appendix B. The essence of the extension is that in the PEC setting the adversary is given access to a combined oracle $(Enc_K + C)(\cdot)$ which first encrypts the plaintext and then compresses the resultant ciphertext.

Definition 4 (CPA-PEC Indistinguishability Experiment): Consider an encryption scheme (Gen, Enc, Dec) and a PEC scheme (C, D). The *CPA-PEC indistinguishability experiment* $\mathsf{Expt}_{\mathcal{A}}^{\mathsf{cpa-pec}}$ is defined as follows:

1) a key K is generated by running Gen;

[2]This can be done by exhaustive search based on a known plaintext-ciphertext pair or (for suitable plaintext distributions) by decrypting a sequence of encrypted blocks and finding a key which decrypts all blocks to elements in the underlying probability distribution.

2) the adversary $\mathcal{A}$ has oracle access to $(Enc_K + C)(\cdot)$, and queries it with a pair of test plaintexts X_0, X_1 of the same length;
3) the oracle randomly chooses a bit $b \leftarrow \{0, 1\}$ and returns the compressed ciphertext $c \leftarrow (Enc_K + C)(X_b)$, called the challenge, to $\mathcal{A}$;
4) the adversary $\mathcal{A}$ continues to have oracle access to $(Enc_K + C)(\cdot)$ and is allowed to make arbitrary queries. Ultimately it makes a guess about the value of b by outputting b';
5) the output of the experiment, $\mathsf{Expt}_{\mathcal{A}}^{\mathsf{cpa-pec}}$, is defined to be 1 if $b' = b$, and 0 otherwise. If $\mathsf{Expt}_{\mathcal{A}}^{\mathsf{cpa-pec}} = 1$, we say that $\mathcal{A}$ succeeded.

Definition 5 (Post-Encryption Compression Security): A PEC scheme (C, D) is called (T, ε)-indistinguishable under chosen plaintext attacks (*CPA-PEC-secure*) if for every adversary $\mathcal{A}$ that runs in time T,

$$\Pr\left[\mathsf{Expt}_{\mathcal{A}}^{\mathsf{cpa-pec}} = 1\right] < \frac{1}{2} + \varepsilon,$$

where the probability is taken over all random coins used by $\mathcal{A}$, as well as all random coins used in the experiment.

We now formulate the security of our CBC compression scheme by the following theorem.

Theorem 6: Let $\mathcal{E} = (Gen, Enc, Dec)$ be a CBC encryption scheme that is (T, ε)-indistinguishable under chosen plaintext attacks, let $\mathcal{P}$ be an efficiently-samplable plaintext distribution, and let (C, D) be a $(\mathcal{E}, \mathcal{P}, \delta)$-PEC scheme. Then (C, D) is $(T/T_C, \varepsilon)$-indistinguishable under chosen plaintext attacks, where T_C is an upper bound on the running time of C.

Proof: Our proof employs a reduction to the security of $\mathcal{E}$. Specifically, we show that if there exists an adversary $\mathcal{A}_C$ that runs in time T/T_C and is able to distinguish between two compressed encryptions, then there exists an adversary $A_{\mathcal{E}}$ that runs in time T and compromises the security of $\mathcal{E}$. Note that the latter implies a break of security of the underlying block cipher using the well-known result by Bellare et al. [14].

Assume, for contradiction, the existence of such an adversary $\mathcal{A}_C$. We construct an adversary $\mathcal{A}_{\mathcal{E}}$ as follows. $\mathcal{A}_{\mathcal{E}}$ invokes $\mathcal{A}_C$ and emulates its oracle $(Enc_K + C)(\cdot)$. That is, for every query X made by $\mathcal{A}_C$, $\mathcal{A}_{\mathcal{E}}$ uses its oracle $Enc_K(\cdot)$ to compute $Enc_K(X)$. Subsequently, it applies the compression algorithm C on $Enc_K(X)$ and forwards $C(Enc_K(X))$ to $\mathcal{A}_C$. When $\mathcal{A}_C$ outputs the two test messages X_0, X_1, $\mathcal{A}_{\mathcal{E}}$ outputs these messages to its own oracle $Enc_K(\cdot)$. Let c^* denote the challenge ciphertext that its oracle returns. $\mathcal{A}_{\mathcal{E}}$ computes $c = C(c^*)$ and forwards c to $\mathcal{A}_C$. When $\mathcal{A}_C$ outputs a bit b', $\mathcal{A}_{\mathcal{E}}$ outputs the same value.

Given that $\mathcal{A}_C$ makes at most T/T_C queries and that the running time of C is upper-bounded by T_C, $\mathcal{A}_{\mathcal{E}}$'s running time is at most T. Let $\frac{1}{2}+\epsilon$ denote the probability that $\mathcal{A}_C$ distinguishes successfully in its game. Clearly, it holds that $\mathcal{A}_{\mathcal{E}}$ distinguishes successfully in its game with the same probability, which is in contradiction to the security of $\mathcal{E}$.

■

V. Public-key Encryption Schemes

After a feasibility study of post-encryption compression in the symmtric key setting, where sender and receiver share a key used for both encryption and decryption, one may wonder whether these results can be extended to the public-key setting. Loosely speaking, in the public-key approach the key generation algorithm produces two different keys (PK,SK) where PK is publicly distributed, yet SK is kept a secret. Thereafter, any party who knows PK can encrypt messages and only the owner of the secret key can decrypt.

Since public key operations for known public-key schemes are computationally expensive, one typically encrypts streams of data by using a public-key scheme to encrypt a symmetric key (e.g., a AES key) and then using this key with a symmetric cipher to encrypt the data. In this case, it is clear that the results in this paper can be applied to the symmetric (bulk) encryption of the data. Yet, the question remains of whether we can compress the PK part of the ciphertext. For the reader familiar with the El Gamal cryptosystem [15], we note that one can compress one of the two components of the ciphertext if, for example, one uses XOR to combine the ephemeral El-Gamal key with the message. In the case of El Gamal over elliptic curves another well-known technique is "point compression" [16] which reduces the ciphertext to half its length. Yet, one could hope for better post-encryption compression (in particular, given that often the ciphertexts are longer than the plaintext). One result in this direction is Gentry's technique for compressing Rabin's ciphertexts [17]. However, this technique can be applied *before* encryption, not post-encryption. Interestingly, the related result by Gentry regarding compression of Rabin's signatures does allow for compression *after* the signature generation, without the need for the signing key.

In conclusion, the question of post-encryption compressibility of ciphertexts produced by public-key schemes is essentially unresolved.

VI. Compression Performance

Codes that are used to compress stream ciphers can be chosen to have arbitrary block lengths, since the method itself does not directly impose any constraints on the block length. On the other hand,

the compression methods that we propose for CFB and CBC mode must operate block-wise since decompression occurs serially and depends upon a previously decompressed block. In effect, the block length of Slepian-Wolf codes must be m. We choose AES as a representative of a widely used block cipher and present our results under assumption that AES is used as the encryption scheme.

The efficiency of Slepian-Wolf compression depends on the performance of underlying Slepian-Wolf codes. It was shown in [18], [19] that Slepian-Wolf compression approaches entropy with speed $O(\sqrt{\frac{\log n}{n}})$, which is considerably slower than $O(\frac{1}{n})$ of arithmetic coding. It follows that for efficient Slepian-Wolf compression, the block length of Slepian-Wolf codes must be long.

Slepian-Wolf codes over finite block lengths have nonzero frame-error rates (FER), which implies that the receiver will sometimes fail to recover $E_K(X_{i-1})$ correctly. Such errors must be dealt with on the system level, as they can have catastrophic consequences in the sense that they propagate to all subsequent blocks. In the following it is assumed that as long as the FER is low enough, the system can recover efficiently, for instance by supplying the uncompressed version of the erroneous block to the receiver. The compression performance depends on the target FER.

We consider a binary i.i.d source with a probability distribution $\Pr(X = 1) = p$ and $\Pr(X = 0) = 1-p$ which produces plaintext bits. A sequence of plaintext bits is divided into blocks of size m, which are encrypted and compressed as described in Section III. The receiver's task is to reconstruct $E_K(X_{i-1})$ from $\tilde{X}_i$ and side information $C(E_K(X_{i-1}))$. For the source considered, Slepian-Wolf decoding is equivalent to error-correction over a binary symmetric channel (BSC), thus the underlying codes should yield a FER that is lower or equal to the target FER over the BSC. Compression efficiency can be evaluated in two ways:

a) fix p and determine the compression rate of a Slepian-Wolf code that satisfies the target FER;
b) pick a well-performing Slepian-Wolf code and determine the maximum p for which target FER is satisfied.

We use low-density parity-check (LDPC) codes [20], which are known to be very powerful and we evaluate compression performance according to method b). In our simulations we used two LDPC codes[3]: the first yields compression rate 0.5 and has degree distribution $\lambda(x) = 0.3317x + 0.2376x^2 + 0.4307x^5$, $\rho(x) = 0.6535x^5 + 0.3465x^6$ and the second yields compression rate 0.75 and degree distribution $\lambda(x) = 0.4249x + 0.0311x^2 + 0.5440x^4$, $\rho(x) = 0.8187x^3 + 0.1813x^4$. All codes were constructed with the Progressive Edge Growth algorithm [21], which is known to yield good performance at short block

[3]The degree distributions were obtained at http://lthcwww.epfl.ch/research/ldpcopt/

lengths. Belief propagation [20] is used for decoding and the maximum number of iterations was set to 100.

p	Target FER	Compression Rate	Source Entropy
0.026	10^{-3}	0.50	0.1739
0.018	10^{-4}	0.50	0.1301
0.068	10^{-3}	0.75	0.3584
0.054	10^{-4}	0.75	0.3032

TABLE I

ATTAINABLE COMPRESSION RATES FOR $m = 128$ BITS.

Simulation results for block length, m, 128 and 1024 bits are shown in Tables I and II, respectively. First, consider the current specification of the AES standard [22], where m is 128 bits. At FER of 10^{-3} the maximum p that can be compressed to rate 0.5 is 0.026, while the entropy for that p is 0.1739. The large gap is a consequence of a very short block length. Notice that for higher reliability, i.e. lower FER, the constraints on the source are more stringent for a fixed compression rate.

p	Target FER	Compression Rate	Source Entropy
0.058	10^{-3}	0.50	0.3195
0.048	10^{-4}	0.50	0.2778
0.134	10^{-3}	0.75	0.5710
0.126	10^{-4}	0.75	0.5464

TABLE II

ATTAINABLE COMPRESSION RATES FOR $m = 1024$ BITS.

When the m is increased to 1024 bits (see Table II) the improvement in performance is considerable. For instance, at FER $= 10^{-3}$ and compression rate 0.5, the source can now have p up to 0.058. In future block-cipher designs one could consider larger blocks, in particular as a way to allow for better post-encryption compression.

VII. CONCLUSION

We considered compression of data encrypted with block ciphers without knowledge of the key. Contrary to a popular belief that such data is practically incompressible we show how compression can

be attained. Our method is based on Slepian-Wolf coding and hinges on the fact that chaining modes, which are widely used in conjunction with block ciphers, introduce a simple symbol-wise correlation between successive blocks of data. The proposed compression was shown to preserve the security of the encryption scheme. Further, we showed the existence of a fundamental limitation to compressibility of data encrypted with block ciphers when no chaining mode is employed.

Some simulation results are presented for binary memoryless sources. The results indicate that, while still far from theoretical limits, considerable compression gains are practically attainable with block ciphers and improved performance can be expected as block sizes increase in the future.

APPENDIX A
COMPRESSING OFB AND CFB MODE

The main part of the paper describes how compression can be performed on data encrypted with block ciphers when they operate in the CBC mode. This section outlines how compression is achieved in other two common modes associated with block ciphers, output feedback (OFB) and cipher feedback (CFB).

A. Output Feedback (OFB)

The mode depicted in Figure 7 is called output feedback (OFB). Plaintext blocks in $\mathbf{X}^n$ are not directly encrypted with a block cipher. Rather, the block cipher is used to sequentially generate a sequence of pseudorandom blocks $\tilde{\mathbf{K}}^n = \{\tilde{K}_i\}_{i=1}^n$ which serve as a one-time pad to encrypt the plaintext blocks. At the output of the encryption algorithm we have $Enc_K(\mathbf{X}^n) = (\text{IV}, \mathbf{Y}^n)$.

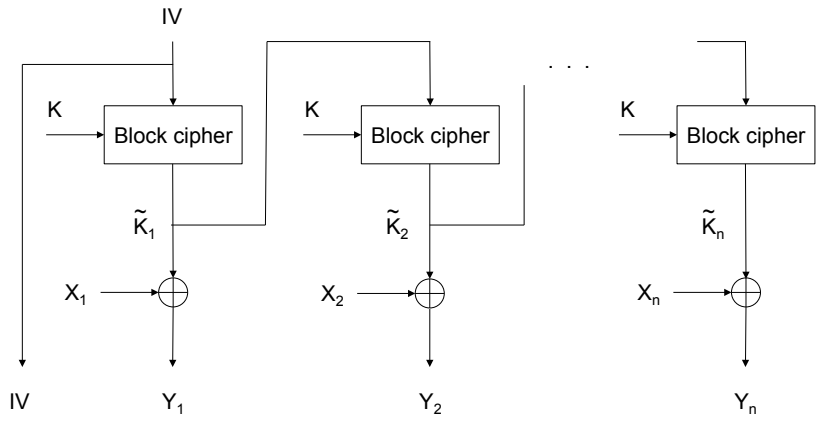

Fig. 7. Output feedback (OFB).

Notice that each block $\tilde{K}_i$ is independent of plaintext blocks, therefore the OFB mode is analogous to the one-time pad encryption scheme and $\mathbf{Y}^n$ can be compressed in the same manner as described in [3]. Most importantly, $\tilde{\mathbf{K}}^n$ and $\mathbf{X}^n$ are independent, therefore the block length of Slepian-Wolf codes that are used to compress $\mathbf{Y}^n$ can be chosen arbitrarily. The IV is uniformly distributed and thus incompressible.

Formally, the compression and decompression algorithms can be described as follows. Let $(C_{\text{OFB}}, D_{\text{OFB}})$ denote a Slepian-Wolf code with encoding rate R and block length nm. The Slepian-Wolf encoding function is defined as $C_{\text{OFB}} : (\mathcal{X}^m)^n \to \{1, \ldots, 2^{mnR}\}$, and the Slepian-Wolf decoding function as $D_{\text{OFB}} : \{1, \ldots, 2^{mnR}\} \times (\mathcal{X}^m)^n \to (\mathcal{X}^m)^n$. Given an output sequence from the encryptor $Enc_K(\mathbf{X}^n) = (\text{IV}, \mathbf{Y}^n)$, compression is achieved by applying the Slepian-Wolf encoder function C_{OFB} to $\mathbf{Y}^n$, so that at the output we have $(\text{IV}, C_{\text{OFB}}(\mathbf{Y}^n))$. Note that C_{OFB} does not require knowledge of K.

Decompression and decryption are performed jointly. The receiver has $(\text{IV}, C_{\text{OFB}}(\mathbf{Y}^n))$ and knows the secret key K, thus it can generate the sequence of pseudorandom blocks $\tilde{\mathbf{K}}^n$ using the IV. Subsequently, it applies the Slepian-Wolf decoder function D_{OFB} to $(C_{\text{OFB}}(\mathbf{Y}^n), \tilde{\mathbf{K}}^n)$ to recover $\mathbf{Y}^n$. The original sequence of plaintext blocks $\mathbf{X}^n$ then equals $\tilde{\mathbf{K}}^n \oplus \mathbf{Y}^n$.

By the Slepian-Wolf theorem, the compression rate approaches entropy of the source asymptotically in nm. That is, even if m is finite, entropy can still be achieved if $n \to \infty$.

B. Cipher Feedback (CFB)

Next, we discuss the cipher feedback (CFB) mode, which is depicted in Figure 8. Similarly as in OFB, the plaintext blocks are not subject to block cipher encryption and $\mathbf{Y}^n$ is obtained by XOR-ing plaintext blocks $\mathbf{X}^n$ with pseudorandom blocks $\tilde{\mathbf{K}}^n$. At the output of the encryption algorithm we have $Enc_K(\mathbf{X}^n) = (\text{IV}, \mathbf{Y}^n)$.

However, in CFB mode, the pseudorandom blocks $\tilde{\mathbf{K}}^n$ are not independent of $\mathbf{X}^n$. Let the block cipher using a secret key K be characterized by the bijective mapping $B_K : \mathcal{X}^m \to \mathcal{X}^m$. Each block $\tilde{K}_i$ depends on the preceding plaintext block X_{i-1} as follows:

$$\tilde{K}_i = B_K(X_i) = B_K(X_{i-1} \oplus \tilde{K}_{i-1}), \tag{4}$$

where X_0 is defined to be the IV.

Due to the dependence between $\tilde{\mathbf{K}}^n$ and $\mathbf{X}^n$, the proposed compression algorithm for the CFB mode operates on individual ciphertext blocks Y_i, rather then on $\mathbf{Y}^n$ all at once like in the OFB mode. Without this distinction, the joint decompression and decryption as proposed in [3] would not be possible. Let

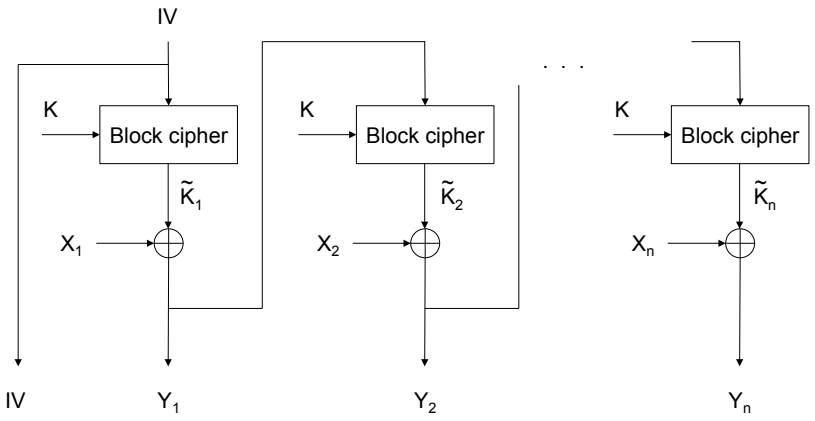

Fig. 8. Cipher feedback (CFB).

$(C_{\text{CFB}}, D_{\text{CFB}})$ denote a Slepian-Wolf code with encoding rate R and block length m. The Slepian-Wolf encoding function is defined as $C_{\text{CFB}} : \mathcal{X}^m \to \{1, \ldots, 2^{mR}\}$, and the Slepian-Wolf decoding function as $D_{\text{CFB}} : \{1, \ldots, 2^{mR}\} \times \mathcal{X}^m \to \mathcal{X}^m$. Compression is achieved by applying the Slepian-Wolf encoding function C_{CFB} to each of the ciphertext blocks in $\mathbf{Y}^n$ individually. The compressed representation of $Enc_K(\mathbf{X}^n)$ is then $(\text{IV}, C_{\text{CFB}}(Y_1), \ldots, C_{\text{CFB}}(Y_n))$. Note again, that C_{CFB} does not require knowledge of K.

Joint decompression and decryption, depicted on Figure 9 is performed sequentially from left to right, since decryption of the ith ciphertext block requires knowledge of the $(i-1)$th plaintext block. Initially, IV is mapped to $\tilde{K}_1$ by the block cipher. The Slepian-Wolf decoder function D_{CFB} is applied to $(C_{\text{CFB}}(Y_1), \tilde{K}_1)$ to obtain Y_1. The first plaintext can now be obtained: $X_1 = \tilde{K}_1 \oplus Y_1$. Subsequently, Y_1 is mapped to $\tilde{K}_2$. The same process repeats to obtain X_2, and later all remaining plaintext blocks $\{X_i\}_{i=3}^n$.

Notice that in contrast to the OFB mode, this compression scheme is generally optimal only if $m \to \infty$. For finite m, the compression is generally suboptimal, even if $n \to \infty$.

APPENDIX B
SOME DEFINITIONS FROM CRYPTOGRAPHY

For completeness, we recall some standard definitions from cryptography that are used in this paper. See [8] for details. These definitions assume a fixed computational model over which time complexity is defined.

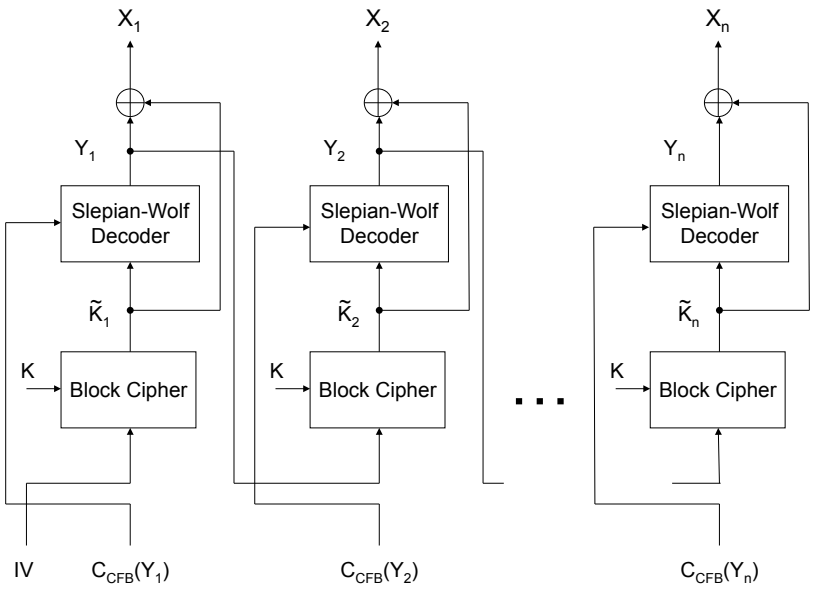

Fig. 9. Joint decryption and decoding in the CFB mode at the receiver is performed serially from left to right.

Definition 7 (Block Cipher): A block cipher B with block size m is a keyed family $\{B_K\}_{K\in\mathcal{K}}$ where for each K, B_K is a permutation over m bits[4] and $\mathcal{K}$ is the set of all possible keys.

The security of a block cipher is defined via the notion of *indistinguishability*. Ideally, we would like the behavior of a block cipher to be indistinguishable by computational means from that of a purely random permutation over m bits. However, since a block cipher is a much smaller family of permutations than the family of *all* permutations, the above is not fully achievable. Yet, if we restrict our attention to "computationally feasible distinguishers" then we can obtain a meaningful notion of security applicable to actual block ciphers such as AES.

Hence, the main ingredient in such definition is that of a *distinguisher*. A distinguisher $Dist$ is defined as a randomized algorithm with oracle access to two m-bit permutations Enc and Dec where $Dec = Enc^{-1}$. $Dist$ can make arbitrary queries to the oracles and eventually outputs a single bit 0 or 1. We consider the runs of $Dist$ in two cases: When the oracles are instantiated with a truly random permutation and when instantiated with a block cipher (i.e., with the functions B_K and B_K^{-1} where K is chosen with uniform probability from the set $\mathcal{K}$). Let P_{REAL} be the probability that $Dist$ outputs a 1 when the oracle was instantiated with B, where P_{REAL} is computed over all random coins of $Dist$ and all choices of the key for B. Further, let P_{RP} be the probability that $Dist$ outputs a 1 when the oracle was instantiated

[4]By a *permutation over m bits* we mean a deterministic bijective function over $\{0,1\}^m$.

with a random permutation, where P_{RP} is computed over all random coins of $Dist$ and all permutations Enc. Intuitively, we can think of $Dist$ as trying to decide if the oracles are instantiated with a random permutation or with a block cipher; hence, a distinguisher is considered successful if the difference $|P_{REAL} - P_{RP}|$, which is referred to as the advantage, is non-negligible. Formally, this leads to the following definition.[5]

Definition 8 (Block Cipher Security): A block cipher B is called (T, ε)-secure if no distinguisher $Dist$ that runs in time T has advantage larger than ε.

This definition tries to capture the idea that, for secure block ciphers, even distinguishers that have the ability to run for extremely large time T, say $T = 2^{80}$, gain only negligible distinguishing advantage, say $\varepsilon = 2^{-40}$. In other words, for any practical purpose the quality of the block cipher is as good as if it was instantiated by a "perfect cipher" (purely random permutation).

A secure block cipher by itself does not constitute a secure private-key encryption scheme due to its deterministic nature. Namely, two identical plaintexts are mapped into two identical ciphertexts, therefore valuable information about data patterns can be leaked to eavesdroppers. Rather, secure block ciphers are used as building blocks that can be used to construct private-key encryption schemes that eliminate this vulnerability. Note that a secure private-key encryption scheme must be probabilistic or stateful.

Toward a formal definition of a secure encryption scheme, consider the following experiment:

Definition 9 (CPA Indistinguishability Experiment): Consider an adversary $\mathcal{A}$ and an encryption scheme (Gen, Enc, Dec). The *chosen plaintext attack (CPA) indistinguishability experiment* $\mathsf{Expt}^{\mathsf{cpa}}_{\mathcal{A}}$ is defined as follows:

1) a key k is generated by running Gen;
2) the adversary $\mathcal{A}$ has oracle access to $Enc_K(\cdot)$, and queries it with a pair of plaintexts X_0, X_1 of the same length;
3) the oracle randomly chooses a bit $b \leftarrow \{0, 1\}$ and returns the ciphertext $q \leftarrow Enc_K(X_b)$, called the challenge, to $\mathcal{A}$;
4) the adversary $\mathcal{A}$ continues to have oracle access to $Enc_K(\cdot)$ and is allowed to make arbitrary queries. Ultimately it makes a guess about the value of b by outputting b';
5) the output of the experiment, $\mathsf{Expt}^{\mathsf{cpa}}_{\mathcal{A}}$, is defined to be 1 if $b' = b$, and 0 otherwise. If $\mathsf{Expt}^{\mathsf{cpa}}_{\mathcal{A}} = 1$, we say that $\mathcal{A}$ succeeded.

Definition 10 (Private-Key Encryption Security): A private-key encryption scheme (Gen, Enc, Dec)

[5]This definition corresponds to the notion of strong pseudorandom permutation [14].

is called (T, ε)-indistinguishable under chosen plaintext attacks (or CPA-secure) if for every adversary $\mathcal{A}$ that runs in time T,

$$\Pr\left[\mathsf{Expt}^{\mathsf{cpa}}_{\mathcal{A}} = 1\right] < \frac{1}{2} + \varepsilon,$$

where the probability is taken over all random coins used by $\mathcal{A}$, as well as all random coins used in the experiment.

APPENDIX C
PROOF OF THEOREM 3

The proof of Theorem 3 is by contradiction. We assume a generic PEC scheme (C, D) that departs in some noticeable way (later made more precise by means of the parameter ε) from the exhaustive strategies when it is applied to a block cipher B. Subsequently, we show how to use such a PEC scheme to build a distinguisher $Dist$ that distinguishes with advantage strictly larger than ε between the block cipher B and a random permutation, in contradiction to the security of B.

For simplicity, and without loss of generality, we assume that D does not make redundant queries to Enc or Dec. Namely, no query X to Enc or query Y to Dec is repeated in a run. If X was output by Dec (resp., Y output by Enc) it is not entered into Enc (resp., into Dec). In addition, if the decoded output from D is X, we assume that X was either input to Enc or output by Dec. If D does not follow these rules, it can be modified to do so.

For $X \in_R \mathcal{P}$ [6] and $Y = Enc(X)$, assume that $C(Y)$ is passed as input to D. We say that D decodes correctly if it queries either X from Enc or Y from Dec during its run. In other words, D is not required to have the ability to identify the correct plaintext, which simplifies our presentation without weakening our results. On the contrary, it shows that even if such a relaxed decoding requirement is acceptable, the lower bound we prove still holds.

Finally, note that the formulation of the theorem assumes that the plaintext distribution $\mathcal{P}$ is efficiently samplable. This assumption is used in an essential way in our proof, though the efficiency requirement from the $\mathcal{P}$ sampler is very weak. In addition, we assume that, for a given compressed ciphertext $C(Y)$, one can sample uniformly from the set $\mathcal{Y}_{C(Y)} = C^{-1}(C(Y))$, which is the set of all ciphertexts mapped by C to $C(Y)$. Additional discussion related to this assumption is given after the proof.

The proof of Theorem 3 uses the following lemma.

[6] We use $X \in_R \mathcal{P}$ to denote that X is chosen at random according to the probability distribution $\mathcal{P}$.

Lemma 11: Let T be a time-bound parameter, let B be a (T,ε)-secure block cipher and let $\mathcal{E} = (Gen, Enc, Dec)$ be an encryption scheme, where Gen chooses $K \in_R \mathcal{K}$ and $Enc = B_K$, $Dec = B_K^{-1}$. Let $\mathcal{P}$ be a plaintext distribution samplable in time $T/4$ and let (C, D) be a generic $(\mathcal{E}, \mathcal{P}, \delta)$-PEC scheme. Then either (C, D) runs in time that exceeds $T/4$ (and hence is infeasible[7]) or the following holds:

(i) Let $X \in_R \mathcal{P}$ and $Y = Enc(X)$. Consider a run of D on input $C(Y)$ in which D queries $Enc(X)$, and let X' be a random element drawn from $\mathcal{P}$ independently of X. Then, the probability that D queries $Enc(X')$ before $Enc(X)$ is at least $1/2 - \varepsilon/(1-\delta)$.

(ii) Let $X \in_R \mathcal{P}$ and $Y = Enc(X)$. Consider a run of D on input $C(Y)$ in which D queries $Dec(Y)$, and let Y' be an element drawn uniformly from $\mathcal{Y}_{C(Y)} = C^{-1}(C(Y))$. Then, the probability that D queries $Dec(Y')$ before $Dec(Y)$ is at least $1/2 - \varepsilon/(1-\delta)$.

We first show how the Lemma 11 suffices to prove Theorem 3.

Proof of Theorem 3: For case (i), the probability that X is queried first is within $\varepsilon/(1-\delta)$ of the probability that X' is queried first, where the latter is the probability of querying a plaintext that bears no information (the run of D is independent of X'). Assuming that $\delta < 1/2$, we get $\varepsilon/(1-\delta) < 2\varepsilon$ and since ε is negligible (say 2^{-40}) so is 2ε. This implies a plaintext-exhaustive strategy by D. Similarly, for case (ii), the probability that the value Y is computed first is within $\varepsilon/(1-\delta)$ of the probability that an independent $Y' \in_R \mathcal{Y}_{C(Y)}$ is computed first. This implies a ciphertext-exhaustive strategy by D. ■

Proof of Lemma 11: First, consider the error-free case, i.e. $\delta = 0$. We show that if (C, D) runs in time less than $T/4$ and conditions (i), (ii) do not hold, we can build a distinguisher (see Appendix B) that runs in time at most T and distinguishes between B and a random permutation with an advantage larger than ε, in contradiction to the security of B. A distinguisher interacts with oracles Enc and Dec and its goal is to identify whether the oracles are instantiated with the real block cipher B or with a random permutation. For clarity, we represent the output 0 from $Dist$ by the symbol RP ($Dist$ decided that the oracles are instantiated with a random permutation) and the output 1 by REAL ($Dist$ decided that the oracles are instantiated by the block cipher B).

We build a distinguisher $Dist$ that uses the scheme (C, D) and responds to the encryption and decryption queries made by D with its Enc/Dec oracles. When Enc is a random permutation, (C, D) may run much longer than when Enc is B. Thus, to bound the time complexity of $Dist$, we set a time

[7]For secure block ciphers, a distinguisher should not be able to attain more then a negligible advantage even if it runs for extremely large time T, say $T = 2^{80}$ (see Appendix B). Thus, a PEC scheme that runs in time $T/4$ would be considered infeasible.

limit $T' = T/4$, such that if the total time of (C, D) exceeds T', $Dist$ stops as well. As we show below, the parameter T' is chosen as $T/4$ to ensure that the total running time of $Dist$ is no more than T.

Initially, $Dist$ chooses $X \in_R \mathcal{P}$ and receives the value $Y = Enc(X)$ from its Enc oracle. It computes $C(Y)$ and passes it as input to D. In addition, $Dist$ chooses an independent $X' \in_R \mathcal{P}$ and independent $Y' \in_R \mathcal{Y}_{C(Y)}$. It is assumed that Y' can be sampled within time $T/4$. Subsequently, $Dist$ monitors the queries to Enc/Dec as requested by D and reacts to the following events:

1) if X is queried from Enc, stop and output REAL;
2) if X' is queried from Enc, stop and output RP;
3) if Y is queried from Dec, stop and output REAL;
4) if Y' is queried from Dec, stop and output RP.
5) if the run of (C, D) exceeds T', stop and output RP

It is possible that multiple such events take place in one run of D, for instance both X and X' may be queried from Enc. In such case $Dist$ stops as soon as it identifies first such event.

Let P_{REAL} and P_{RP} be defined as in Appendix B. We evaluate the advantage of $Dist$, namely, the difference $|P_{REAL} - P_{RP}|$. First, consider a run of $Dist$ when Enc/Dec are instantiated by a random permutation. The behavior of (C, D) depends on Y which is chosen at random and independently of X and X', therefore the run is independent of both X and X'. In effect, the probability that X is queried before X' is exactly 1/2. Similarly, if Y is queried from Dec, the behavior of D depends on $C(Y)$, while both Y and Y' have the same probability to be the chosen as the preimage of $C(Y)$. Therefore, the probability that Y precedes Y' is exactly 1/2. It follows that $Dist$ outputs REAL with probability at most 1/2 (exactly 1/2 for the X and Y cases and with probability 0 if $Dist$ exceeds time T'), i.e., $P_{RP} \leq 1/2$.

Now, consider a run of $Dist$ when Enc/Dec are instantiated by a block cipher B and its inverse, respectively. Assume, for contradiction, that (C, D) stops before time T' and either (i) the probability that X' is queried before X is strictly less than $1/2 - \varepsilon$ or (ii) the probability that Y' is queried before Y is strictly less than $1/2 - \varepsilon$. It follows that the probability that $Dist$ outputs RP is strictly smaller than $1/2 - \varepsilon$, therefore $P_{REAL} > 1/2 + \varepsilon$.

Thus, $Dist$ distinguishes with advantage $|P_{REAL} - P_{RP}|$, which is strictly larger than ε. The running time of $Dist$ is upper-bounded by T: it includes three samplings (of X, X' and Y'), each assumed to take at most $T/4$ time, and the work of (C, D) which $Dist$ runs for total time $T/4$ at most. In all, we have built a distinguisher against B that runs time T and has advantage larger than ε in contradiction to the security of the block cipher B.

Assume now that $\delta > 0$. When a positive probability of error for D is allowed, it can occur that D stops before time T' and before $Dist$ sees X, X', Y or Y'. To deal with this situation, we add a clause to the specification of $Dist$ saying that if (C, D) stops before time T' and before seeing any of the values X, X', Y, Y', then $Dist$ chooses a random bit b and outputs REAL if $b = 1$ and RP if $b = 0$. We slightly increased the probability that X' is queried before X (or Y' before Y) in the block cipher case, however it is still negligibly far from $1/2$. The proof is now a straightforward extension of the case when $\delta = 0$. ■

Remark. The proof of Lemma 11 assumes that the set $\mathcal{Y}_{C(Y)}$ is samplable in time $T/4$. While this assumption is likely to hold, we note that it is enough to know the size of $\mathcal{Y}_{C(Y)}$. In such case, rather than sampling $\mathcal{Y}_{C(Y)}$, the events 3) and 4) in the proof are replaced with the following one: if the number of queries to Dec performed by D exceeds $|\mathcal{Y}_{C(Y)}|/2$ before Y is queried, stop and output RP.

We now sketch the proof of the theorem when none of the above conditions holds, namely when $\mathcal{Y}_{C(Y)}$ is of unknown size and not samplable in time $T/4$. Assume that (C, D) performs noticeably better than the exhaustive strategies when the Enc/Dec oracles are instantiated with the block cipher B. Then it must hold that (C, D) runs noticeably faster when the Enc and Dec oracles are instantiated with the block cipher B than with a random permutation, for the exhaustive strategies are optimal for a random permutation (as shown above). Using this (assumed) discrepancy between the runs of (C, D) over B and the runs of (C, D) over a random permutation, we can build a distinguisher against B in contradiction to the security of B. In the following, we formalize this discrepancy and outline the construction of the distinguisher, where some straightforward details are omitted.

For any plaintext X let $T_B(X)$ denote the runtime of (C, D) on input X when Enc/Dec oracles are instantiated with the block cipher B. Further, let $T_R(X)$ denote the runtime of (C, D) on input X when Enc/Dec oracles are instantiated with a random permutation. We assume that there is a known time bound T_0 such that $T_B(X) < T_0$ for all X (we relax this assumption below) and there exists a non-negligible ε such that $\Pr[T_B(X) < T_R(X)] \geq 1/2 + \varepsilon$. The probability is over all choices of X and all random coins of (C, D). Further, it is over all key choices for B for T_B and over all random permutations for T_R. We build a distinguisher $Dist$ as follows:

1) Choose $X \in_R \mathcal{P}$ and run (C, D) using the input oracles. If time T_0 is exceeded, stop and output RP, otherwise proceed to Step 2.
2) Let T_1 denote the running time of (C, D) in step (1). Run (C, D) again on X (same X as in step 1)) but this time ignore the given Enc/Dec oracles. Instead, answer queries from (C, D) with a

random permutation. If time T_1 is exceeded then output REAL, otherwise output RP.

We have the following:

- If the Enc/Dec oracles are instantiated with B, step (1) always completes and in step (2) REAL is output with the probability that equals $\Pr[T_R(X) > T_B(X)]$, which is at least $1/2 + \varepsilon$.
- If the Enc/Dec oracles are instantiated with a random permutation, REAL is output only if $T_R(X) \leq T_0$ and the runtime on X in step (2) exceeds the runtime on X in step (1). The probability of the latter is at most 1/2 since both runs are over a random permutation.

Thus, $Dist$ will output REAL with probability at least $1/2 + \varepsilon$ when the oracles are instantiated with B, while it will output REAL with probability at most $1/2$ when the oracles are instantiated with a random permutation. It follows that $Dist$ is a (T, ε)-distinguisher for B where $T = 2T_0$ since in each of the steps (1) and (2) $Dist$ runs time at most T_0.

Note that the requirement that $T_B(X) < T_0$ for all X can be relaxed such that the joint probability of $T_B(X) < T_0$ and $T_B(X) < T_R(X)$ is at least $1/2 + \varepsilon$.

REFERENCES

[1] U. D. of Commerce/National Institute of Standards and Technology, "Advanced encryption standard (AES)," in *FIPS PUB 197*, Nov. 2001.

[2] N. B. of Standards, *Data Encryption Standard (DES)*. U.S. Department of Commerce, Washington D.C, 1977.

[3] M. Johnson, P. Ishwar, V. Prabhakaran, D. Schonberg, and K. Ramchandran, "On compressing encrypted data," *IEEE Trans. Signal Processing*, vol. 52, pp. 2992–3006, Oct. 2004.

[4] D. Slepian and J. Wolf, "Noiseless coding of correlated information sources," *IEEE Trans. Info. Theory*, vol. 19, pp. 471–480, July 1973.

[5] S. Kent and K. Seo, "Security achitecture for the internet protocol," in *RFC 4301*, Dec. 2005.

[6] T. Dierks and E. Rescorla, "The TLS protocol – version 1.2," in *RFC 5246*, Aug. 2008.

[7] M. Johnson, D. Wagner, and K. Ramchandran, "On compressing encrypted data without the encryption key," in *Proc. of the Theory of Crypto. Conf.*, Cambridge, MA, Feb. 2004.

[8] J. Katz and Y. Lindell, *Introduction to Modern Cryptography*. Chapman & Hall/CRC, 2007.

[9] A. Aaron and B. Girod, "Compression with side information using turbo codes," in *IEEE Data Compression Conf.*, 2002, pp. 252–261.

[10] J. Garcia-Frias, "Compression of correlated binary sources using turbo codes," *IEEE Communications Letters*, vol. 5, pp. 417–419, Oct. 2001.

[11] A. Liveris, Z. Xiong, and C. Georghiades, "Compression of binary sources with side information at the decoder using LDPC codes," *IEEE Communications Letters*, vol. 6, pp. 440–442, Oct 2002.

[12] J. L. Massey, "Guessing and entropy," in *IEEE Inter. Symp. on Info. Theory*, Seattle, WA, June 1994.

[13] C. Cachin, "Entropy measures and unconditional security," Ph.D. dissertation, ETH Zürich, 1997.

[14] M. Bellare, A. Desai, E. Jokipii, and P. Rogaway, "A concrete security treatment of symmetric encryption: Analysis of the DES modes of operation," in *IEEE Proc. of 38th Annual Symp. on Foundations of Computer Science*, 1997.

[15] T. ElGamal, "A public-key cryptosystem and a signature scheme based on discrete logarithms," in *CRYPTO '84*. Springer-Verlag (LNCS 196), 1984, pp. 10–18.

[16] I. F. Blake, G. Seroussi, and N. P. Smart, *Eliptic Curves in Cryptography*. Cambridge University Press, 2000.

[17] C. Gentry, "How to compress Rabin ciphertexts and signatures (and more)," in *CRYPTO '04*. Springer-Verlag (LNCS 3152), 2004, pp. 179–200.

[18] D. He, L. Lastras-Montaño, and E. Yang, "A lower bound for variable rate Slepian-Wolf coding," in *IEEE Inter. Symp. on Info. Theory*, Seattle, WA, July 2006.

[19] ——, "On the relationship between redundancy and decoding error in Slepian-Wolf coding," in *IEEE Information Theory Workshop*, Chengdu, China, Oct. 2006.

[20] T. Richardson and R. Urbanke, *Modern Coding Theory*. Cambridge University Press, 2008.

[21] X. Hu, E. Eleftheriou, and D. Arnold, "Regular and irregular progressive edge-growth Tanner graphs," *IEEE Transactions on Information Theory*, vol. 51, no. 1, pp. 386–398, Jan 2005.

[22] W. Mao, *Modern Cryptography: Theory and Practice*. Prentice Hall, 2003.