Secure Data Communication using Data Encrypted with Block Ciphers

Aswathy Venu Gopalan
PG Scholar, Dept. of ECE
Muslim Association College of Engineering,
Trivandrum, India

Mr. Ajith S S
Assistantprofessor, Dept. of ECE
Muslim Association College of Engineering,
Trivandrum, India

***Abstract*—Traditionally in communication systems, data from a source are first compressed and then encrypted before transmission over a channel to the receiver. Data compression is known for reducing storage and communication costs. It involves transforming data of a given format, called source message to data of a smaller sized format called codeword. Data encryption is known for protecting formation from eavesdropping. It transforms data of a given format, called plaintext, to another format, called cipher text, using an encryption key. This paper investigates compression of data encrypted with block ciphers, such as the Advanced Encryption Standard. It is shown that such data can be feasibly compressed without knowledge of the secret key. 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. Combined decryption and decompression is also performed to reduce decryption failure.**

Index Terms —Block ciphers, cipher block chaining (CBC) mode, compression, electronic code book (ECB) mode, encrypted data

I. INTRODUCTION

Usually in communication systems, data from a source are first compressed and then encrypted before transmission over a channel to the receiver. In this paper the problem of compressing encrypted data is considered. While in many cases the traditional approach is befitting, there exist scenarios where there is a need to reverse the order in which data encryption and compression are performed. For example, consider 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 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) and Data Encryption Standard (DES). 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, input and output have the same entropy. 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. It was believed that encrypted data is practically incompressible. Compression is practically achievable due to a simple symbol-wise correlation between the key (one-time pad) and the encrypted
message.By using standard techniques in the cryptographic literature it proves that the proposed compression schemes do not compromise the security of the original encryption scheme. It also shows 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.

II. PRELIMINARIES

We begin with a standard formal definition of an encryption
scheme . A private-key encryption scheme$Enc_K(X)$; is a triple of algorithms (*Gen*,*Enc*,*Dec*) , where *Gen* is a probabilistic algorithm that outputs a key 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 takes as input a key*K*and a ciphertext and outputs a plaintext$Dec_K(Enc_K(X)) = X$; . It is required that for every key output by *Gen*and every plaintext*X*, we have$Dec_K(Enc_K(X)) = X$.
In private-key encryption schemes concerned in this paper the same key is used for encryption and decryption algorithms. Private-key encryption schemes are divided in two categories: stream ciphers and block 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 accomplish encryption by means of nonlinear mappings on input blocks of fixed length; common examples are AES and DES. Block ciphers are typically 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.

We are interested in systems that perform both compression
and encryption, wherein the compressor has no access to the
key. In such systems encryption is performed after compression. This is a consequence of the

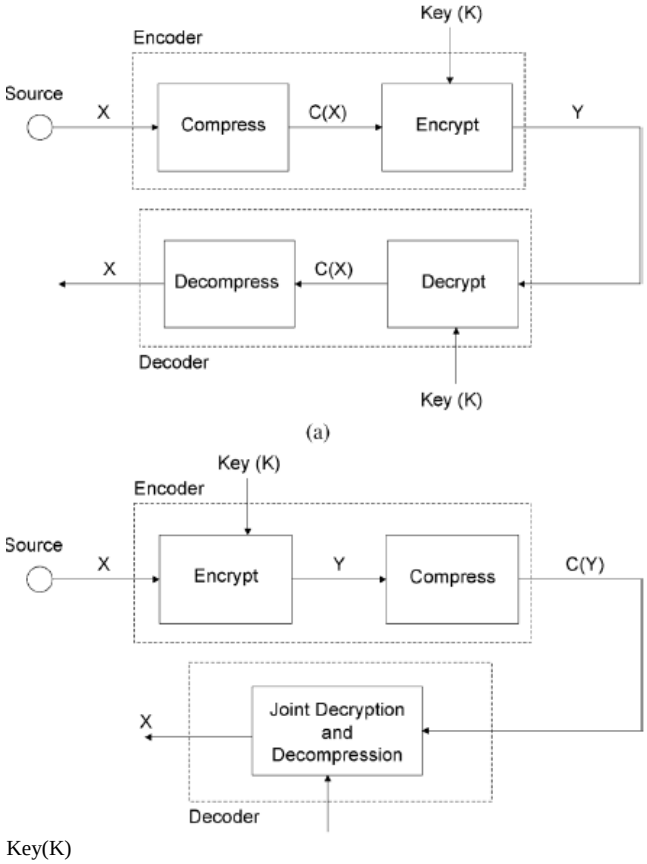

Fig. 1. Systems combining compression and encryption. (a) Traditional system with compression preceding encryption. (b) System with encryption subsequent to compression

traditional view which considers encrypted data hard to compress without knowledge of 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* using

$$Y_i \triangleq X_i \oplus K_i$$

This is followed by compression, which is agnostic of K , to generate the compressed ciphertext .

III. COMPRESSING BLOCK-CIPHER ENCRYPTION

As opposed 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 decipher 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 from previous encryptor inputs or outputs. This randomization prevents identical plaintext blocks from being encrypted into identical ciphertext blocks.

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.

A. *CBC*

The most common mode of operation is CBC. Depicted inFig. 2 block ciphers in CBC mode are employed as the default mechanism in widespread security standards such as IPSec and TLS/SSL and hence it is a common method of encrypting internet traffic.

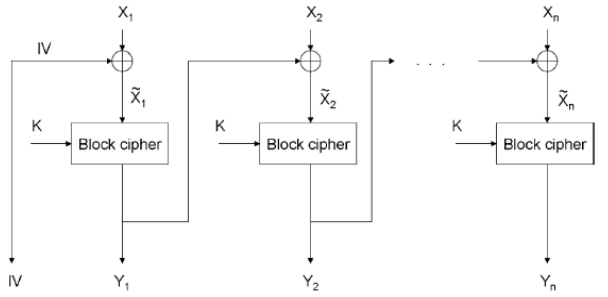

Fig. 2 CBC

Fig. 3 Compressor

In the CBC mode, each plaintext block $X_i$is randomized prior to encryption, by being XOR-ed with the ciphertext block Y_{i-1}corresponding to the previous plaintext block $X_{i-1}$to obtainX_i .The ciphertext block $Y_i$is generated by applying the block cipher with key K to the randomized plaintext block $X_i$according to

$$Y_i = B_K(X_i \oplus Y_{i-1})$$

Where Y_0= IV and $B_K$is the block cipher mappingusing the key. At the output of the encryption algorithm,we have $Enc_K(X^n) = (IV , Y^n)$.

IV. TURBO ENCODER

The turbo encoder consists of two RSC (Recursive Systematic Coders) with an inter-leaver separating them. The block diagram for turbo encoder is shown in figure 4. It is comprised of two transfer functions which represent the systematic components of RSC encoders and an inter-leaver. The input symbols are simply permuted in a random fashion and forwarded to the second RSC encoder. RSC encoders work best due to their IIR response. The transfer function of each encoder
is given by,

$$Z(x) = h(x)/g(x)$$

It is important for both encoders to have the same transfer function.

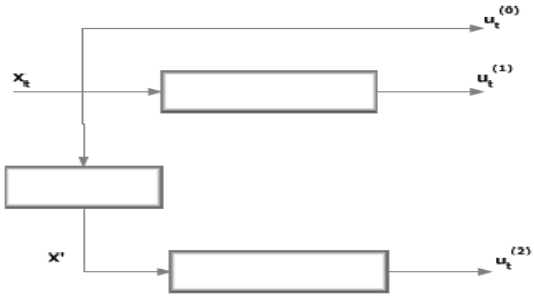

Fig.4 Turbo encoder

V. TURBO DECODER

The block diagram of turbo decoder is shown in figure 5After modulation, the encoded data is transmitted through a communication channel and the output of the channel is a vector *v*.The vector *v* is first de-multiplexedinto the vectors $V^{(0)}$(corresponding to $U^{(0)}$), $V^{(1)}$(corresponding to $U^{(1)}$ and $V^{(2)}$(corresponding to $U^{(2)}$). The working of the decoder is as follows; the data$V^{(0)}$, $V^{(1)}$related to first encoder are fed to Decoder 1. The algorithm running inside decoder 1 is Viterbi algorithm explained in next section. The output of decoder 1 is interleaved and then fed to decoder 2. As the data from RSC encoder 2 was the interleaved version of the input, this restricts the vector $V^{(0)}$to be passed through an interleaver before passing through decoder 2. Then, the output from decoder 2 is fed to decoder 1 and this process of transferring the information back and forth continues until some maximum number of iterations is achieved.

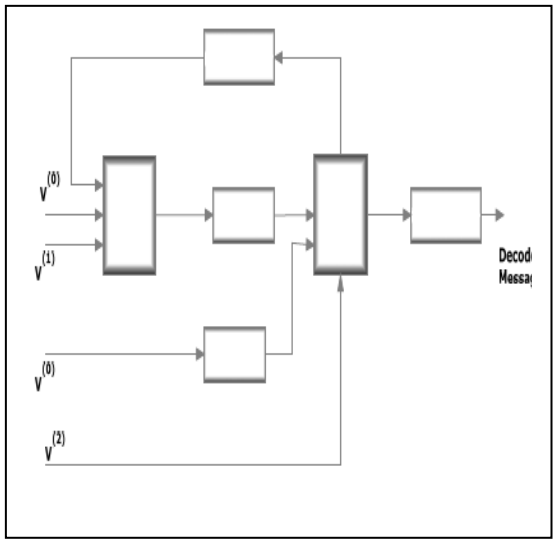

Fig 5 . Turbo decoder

VI. AWGN CHANNEL

For this channel, output from demodulator is a continuous alphabet. We cannot describe these alphabets as correct or incorrect. It is seen that maximizing the $p(z|u^m)$is equal to maximizing the correlation between the codeword$u^m$and the value received from channel z given below:

$$\sum_{i=1}^{\infty}\sum_{f=1}^{\infty} z_{ij}\, u_{ij}^{m}$$

So decoder will select a code-word that provides closest Euclidean distance to z. in order to apply the above equation, decoder needs to be able to handle analog values.But this is impractical because decoders are implemented in digital domain, so we need to quantize our data. This quantized Gaussian channel referred to as soft decision channel.

VII. PUBLIC-KEY ENCRYPTION SCHEMES

After a feasibility study of PEC in private-key encryptionschemes, one may wonder whether these results can be extended to public-key encryption schemes. Loosely speaking, the key generation algorithm in public-key encryption schemes produces two different keys: a public key, known to everyone, and a secret key. Any party who knows the public key can encrypt messages, but only the owner of the secret key can decrypt. Public-key encryption schemes are computationally expensive; therefore, one typically encrypts streams of data in twostages. In the first stage, a public-key encryption scheme is usedto encrypt a private key, and in the second stage, a private-key encryption scheme (using the private key from the first

stage) is used to encrypt the data. The legitimate recipient is assumed tohave the secret key, so it can decrypt the private key and subsequently use the private key to decrypt the data. The results in this paper can be applied to compress data encrypted in the second stage, while the question whether the private key encrypted with a public-key encryption scheme in the first stage can be compressed deserves further study and is beyond the scope of this paper.

TABLE I
COMPRESSION RATES AND HIGHEST ATTAINABLE PROBABILITY p FOR BLOCK LENGTH $m = 128$ bits

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

TABLE II
COMPRESSION RATES AND HIGHEST ATTAINABLE PROBABILITY p FOR BLOCK LENGTH $m = 1024$ bits

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

Simulation results for block lengths of 128 and 1024 bits are
shown in Tables I and II, respectively. First, consider the current specification of the AES standard [23], where m is 128 bits. At FER of10^{-3} the binary source to be compressed to rate 0.5, can have the probability of at most 0.026, where its entropy equals0.1739. The large gap between the achievable compression rate and the entropy is a consequence of the very short block length. Note that for a fixed compression rate the maximum probability p decreases with the decreasing target FER. An increase in block length to 1024 bits results in a considerable improvement in performance (see Table II). For instance, at FER = 10^{-3} and compression rate 0.5, the source can now have probability up to 0.058. In future, block cipher designs one could consider longer block sizes, in order to allow for better PEC.

VIII. CONCLUSION

We considered compression of data encrypted with block ciphers without knowledge of the key. Contrary to the widespread belief that such data are practically incompressible, we show that compression can be attained. Our method hinges on the fact that chainingmodes widely used 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 and by adopting turbo coding bit error rate performance is improved.

IX. REFERENCES

1. Siva ThejaMaguluri,*" Compressing encrypted data"* ECE 859 RB cryptography May 9 , 2009
2. T. Subhamasthan Rao , M. Soujanya , T.Revathy ,T. Hemalatha , *" Simultaneous Data Compression and Encryption "* IJCSIT , vol 2(5) , 2369 - 2374, 2011.
3. Dr.V.KGovindan, B.S Shajee Mohan IDBE,*"An Intelligent Data Encryption and Compression for High Speed and Secure Data Transmission over Internet"*Inter national conference on intelligent signal processing and robotics. IIT Allahabad , February 2004.
4. Sashikala, Melwin .Y., Arunodhayan Sam Solomon, M.N.Nachappa , *"A Survey of Compression Techniques"* International Journal of Recent Technology and Engineering (IJRTE) ISSN: 2277-3878, Volume-2, Issue-1, March 2013
5. M. Johnson, P. Ishwar, V. Prabhakaran, D. Schonberg, and K. Ramchandran,*"On compressing encrypted data,"* IEEE Trans. SignalProcess., vol. 52, no. 10, pp. 2992-3006, Oct. 2004.
6. M. Baritha Begum Y. Venkataramani "*A New Compression Scheme for Secure Transmission*"Vol 10(6), December 2013, 578-586
7. D. Schonberg,S. C. Draper,K. Ramchandran, *"On Blind Compression of Encrypted Correlated Data Approaching The Source Entropy Rate"*.EECS Dept. of California, Berkeley, CA 947-1772.
8. VijaysinhPatil,P.C. Latane, Lonavala*"Implementation Of Efficient Turbo Code Encoder- Decoder With Max-Log-Map Algorithm"* International Journal of Advanced Technology & Engineering Research (IJATER),2014
9. Sayantan Choudhury *"Modeling and Simulation of a Turbo Encoderand Decoder for Wireless CommunicationSystems"* May 2007.
10. M. Johnson, D. Wagner, and K. Ramchandran, *"On compressingencrypted data without the encryption key,"* presented at the TheoryCrypto. Conf., Cambridge, MA, Feb. 2004.
11. Aaron and B. Girod, *"Compression with side information usingturbo codes,"* in Proc. IEEE Data Compress. Conf., 2002, pp. 252–261.
12. J. Garcia-Frias, *"Compression of correlated binary sources using turbocodes,"* IEEE Commun. Lett., vol. 5, no. 10, pp. 417–419, Oct. 2001.
13. D.A. Huffman, *"A Method for the Construction of Minimum-Redundancy Codes"*, in: Proc. IRE, 40, 9 (Sept.),1952, pp.
14. I.H. Willen, RandfordM.Neala& John G.Cleary, *"Arithmetic Coding for DataCompression*", Communications of the ACM Volume 30 Issue 6,1988.
15. Mamta Sharma, *"Compression Using Huffman Coding*" IJCSNS International Journal ofComputer Science and Network Security, VOL.10 No.5, May 2010