Alternative N-bit Key Data Encryption for Block Ciphers

Kayque M. C. Damasceno[1,2], Carlos A. de Moraes Cruz[1], Anderson V. C. de Oliveira[2], Luís S. O. de Castro[1]

[1]Departamento de Eletrônica e Computação – Universidade Federal do Amazonas (UFAM) – Manaus – AM – Brazil

[2]Sidia Instituto de Ciência e Tecnologia – Manaus – AM – Brazil

kayquedamasceno18@gmail.com, agscruz@hotmail.com

anderson.oliveira@sidia.com, luisfne@gmail.com

Abstract. *Post-encryption patterns are cribs that can be used by adversaries to unlock the encryption key both in symmetric or asymmetric cryptography, compromising security. Different methods to mitigate the problem, with their advantages and disadvantages, can be found in the literature, including one-time pad encryption, code-based cryptography and cipher block chaining. This work presents an alternative technique to generate an n-bit, n-block and key cipher that can be derived from reasonable short length key. The proposed technique is able to mitigate post-encryption patterns. Experimental results asserting the capabilities of the proposed cipher techniques are presented and discussed in the text.*

1. Introduction

In the realm of symmetric cryptography and in some methods of asymmetric cryptography, considering that the encryption algorithm is well known to any adversary, the last barrier of security is the privacy of the encryption key [Barker et al. 2007]. For the sake of security, it is expected only the sender and the receiver to have access to the encryption key. This is true for symmetric cryptography and some methods of asymmetric cryptography, such as Diffie-Hellmn and elliptic-curve encryption, but not to RSA encryption. In RSA, the encryption key is public and the private key is employed in the decryption process. Therefore, the encryption key length is currently one of the primary concerns for encryption systems [NSA 2009] and [NSA 2016]. The length of the encryption key determines the necessary time frame for the adversary to crack it using brute force, or trying to solve the discrete logarithm problem (DLP), or elliptic curve DLP (ECDLP) with a determined computation power. The larger the amount of time necessary to crack the encryption key, the more secure the encryption is believed to be. Concerns on the length of encryption keys might become worst as quantum computer technologies knock the door [Sendrier 2017].

Encryption of large amount of data have long been a problem to be dealt with by cryptographers, mainly because of post-encryption pattern (PEP) left behind when short-sized keys compared with the message length are employed. Such patterns are types of cribs that can be used by adversaries trying to decipher the message encryption key. In order to mitigate PEP, different encryption techniques may be applied including one-time pad encryption [Bellovin 2011], [Rijmeanants 2012], code-based cryp-

tography [Tao et al. 2013], [Sendrier 2017], cipher block chaining and its derivatives [Ehrsam et al. 1978], [FIPS 1977], [FIPS 1981], and [Dierks and Rescorla 2006].

One-time pad encryption is probably the best way to avoid PEP, however it is very hard to produce and store keys with the same length of large messages. Code-based encryption is also a promising method to encrypt large amount of data [Tao et al. 2013], [Sendrier 2017], however, as occurs with one-time pad, it uses very large key length, from 100s of kilobytes to several megabytes, what makes key storage a problem for the method. Cipher block chaining and its derivatives such as propagating cipher block chaining and cipher feedback [Ehrsam et al. 1978], [FIPS 1977], [FIPS 1981], and [Dierks and Rescorla 2006] are interesting approaches to mitigate PEP, because they use reasonable short key lengths, compliant with those recommended by [NSA 2009] and [NSA 2016]. However, the need of a source of randomness [Dierks and Rescorla 2006] to produce its initialization vector is an issue to the method.

In this work, an alternative cipher technique to mitigate PEP is proposed. The technique is suitable for n-bits, n-blocks data encryption. It uses neither initialization vector, as in classical cipher block chaining, nor the same key to encrypt each data block of the message. Moreover, following the principles of one-time pad encryption, each message can be said to be encrypted with its own encryption key.

The paper is organized as follows: Section 2 presents the details of the proposed alternative technique. Section 3 discusses the results of some encryptions performed with the proposed technique. Conclusions are summarized in Section 4.

2. The Proposed N-bits Encryption Technique

The pseudo code of the proposed n-bits data encryption method is shown in Table 1.

Table 1. Proposed Encryption Algorithms

Encryption:	
Input:	*MSN == file with nbytes;*
	n == number of blocks of MSN;
	ENCK == encryption key;
Step 1:	*e = HASH(MSN, H0);*
Step 2:	*H1 = HASH(ENCK, H0);*
Step 3:	*BLK[0] = e XOR ENCK;*
Step 4:	*H1 = HASH(e, H1);*
Step 5:	*for (x = 1; x<= n; x++)*
	{
	BLK[x] = blk[x] XOR H1;
	H1 = HASH(H1, H1);
	}
Output:	*BLK[0]...BLK[n];*

The method relies mainly on the privacy of the message encryption key (*ENCK*), which is known only by the sender and the receiver, and on the believed irreversibility of the chosen HASH function. The initialization vector H_0 is a public constant, while H_1 is a dynamic one. In this work, *ENCK* is obtained by multiplying the sender private key with the receiver public key, or by multiplying the receiver private key with the sender public key, using the elliptic curve SECP256K1. The HASH function employed in this work is the SHA256. The minimum message size considered for this method is that of the output of the HASH function, in this case 256 bits.

The proposed encryption method works as follows. Given a message (*MSN*) of n-bytes as input and *ENCK*, where the number of blocks of the message n is determined as the ratio between the size of *MSN* in bits and the size in bits of *ENCK*, the first step to encrypt *MSN* is to take a HASH of it and store into e, as shown in Table 1. In the second step, a HASH of *ENCK* is taken and stored into $H1$. The hash of the *MSN*, e, is then encrypted with *ENCK* and stored into BLK[0]. Here, the encryption is done by xorring a block and the encryption key, where the block must be of the length of the encryption key in bits.

In the fourth step of encryption, a hash of e using $H1$ as the initialization vector of the hash function is taken and stored into $H1$. Then, $H1$ becomes the key to encrypt the first block of *MSN*. Next, all the n blocks of *MSN* are encrypted in step five. Each of the n blocks is encrypted with a different key, the first of which is the result of $H1$ from step four. All the subsequent encryption keys are hashes of the current $H1$ using itself as initialization vector of the employed hash function.

Table 2. Proposed Decryption Algorithms

Decryption:	
Input:	*BLK[0]...BLK[n];*
	ENCK == encryption key;
Step 1:	*H1 = HASH(ENCK, H0);*
Step 2:	*e = BLK[0] XOR ENCK;*
Step 3:	*H1 = HASH(e, H1);*
Step 4:	*for (x = 1; x<= n; x++)*
	{
	blk[x] = BLK[x] XOR H1;
	H1 = HASH(H1, H1);
	}
Output:	*e, blk[1]...blk[n];*
	MSN == blk[1]||blk[2]||...||blk[n];
	//To verify msn integrity:
	e == HASH (msn, H0);

The output of the encryption algorithm is the encrypted blocks from BLK[0] to BLK[n] that must be sent to the receiver. The decryption algorithm is the one shown in

Table 2. To decrypt the message, the receiver uses the *ENCK* employed in the communication and the encrypted blocks received from the sender.

In the first step of decryption, a hash of the *ENCK* is taken and stored into $H1$. Then the hash of the encrypted message, e, is decrypted from BLK[0] by xorring it with *ENCK*. Next, in step three, a hash of e using the current $H1$ as initialization vector of the hash function is taken and stored into $H1$. The new value of $H1$ is the key to decrypt the first block, blk[1], of the message in step four. All the subsequent blocks up to blk[n] are decrypted by xorring each block with their corresponding decryption keys that are hashes of current $H1$ using itself as initialization vector of the employed hash function.

The output of the decryption algorithm is the recovered *MSN* that is restored by appending the decrypted blocks from $blk[1]$ to $blk[n]$. The integrity of the received message can also be verified by taking a hash of *MSN* and comparing it with e.

In order to show how the proposed encryption method is suitable to overcome PEP, some examples of big messages encrypted with the proposed method and with two others are presented in the next section.

3. Experimental Results

The experimental results assert the PEP mitigation ability of the proposed method. In order to visually show the PEP mitigation effect and compare it with other encryption methods susceptible to PEP, four different images are encrypted with the proposed method and two other PEP prone methods. The results show that the simpler the image to be encrypted is, the larger the PEP effects they present.

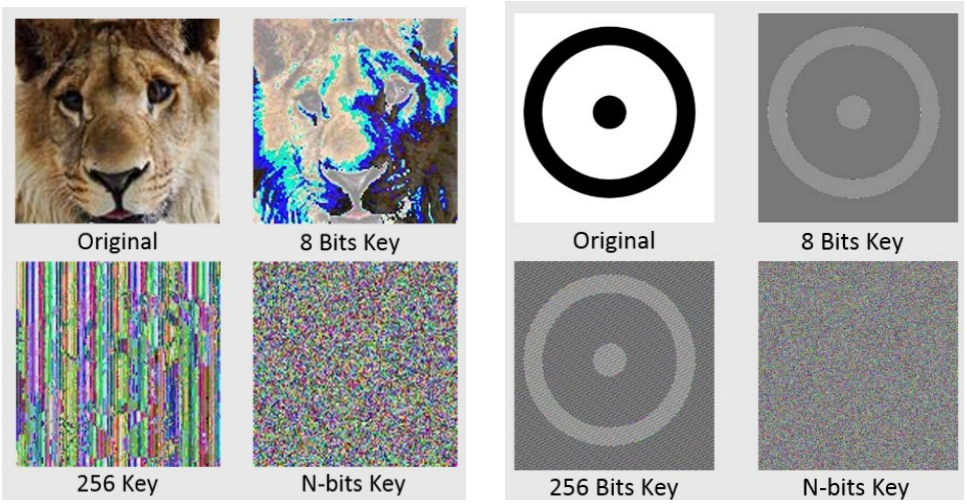

Figure 1. (a) a lion, (b) a target.

The first encryption example is performed over the image of a lion, shown in Figure 1 (a). It is the more complex of the four presented examples, and is performed with an encryption key of only 8 bits, a completely insecure one, where the patterns of the original image is very clear and can be used by an adversary to unlock the encryption key. The second encryption, performed over the same image, employs a key of 256 bits derived from the elliptic curve SECP256K1 that is a still secure key length according

to [NSA 2016]. In this case, the amount of PEP compared with the 8 bits key length encryption is reduced, however a clever adversary still have a lot of cribs to search for the encryption key. The third encryption performed over the image of the lion is done by using the proposed n-bits encryption method. The results of which present no apparent PEP.

The second encryption example is performed over the image of a target, shown in Figure 1 (b). This is a very simple image compared with the lion image, where very large amount of PEP can be observed with both encryption methods of 8 bits and 256 bits. Notwithstanding, the proposed encryption method is shown to hide well the PEP effects.

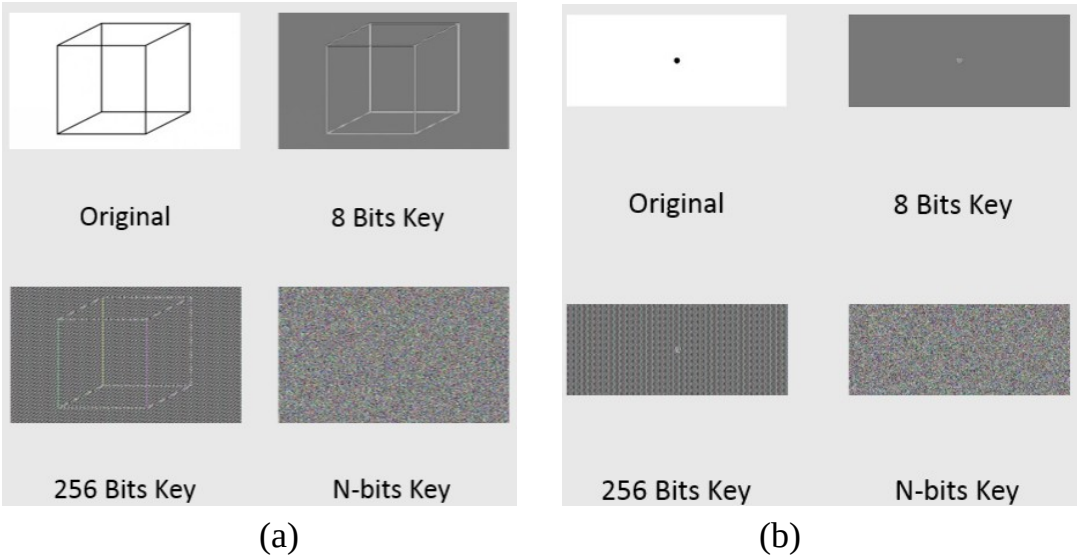

Figure 2. (a) a cube, (b) a point.

The third and fourth encryption examples are performed over the images of a cube and a of point, shown in Figures 2 (a) and (b), respectively. Again, they are very simple images compared with that of the lion. The encryptions with 8 bits and 256 bits also show a lot of PEP effects as that of Figure 1 (b). However, when encrypted with the proposed n-bits method, all the apparent PEP effects are hidden.

The authors understand that in order to assert the real capabilities of the proposed method, as well as its security, more sophisticated tests and formal verification must be performed to determine whether it is cryptographically secure or not. However, the preliminary presented results show that the proposed method is able to hide well visual PEP when compared with simpler encryption methods. As H_0 is a public constant, it is necessary to assert also whether the method is vulnerable to chosen-plaintext attack or not. Future investigations with more sophisticated tests are the next targets of this research, together with a more robust set of comparisons to previous methods presented in the literature.

4. Conclusions

In this work, we have presented an alternative n-bits block cipher encryption method that is shown to successfully overcome post-encryption pattern. Experiments were performed over image files with different complexities. The results show that, while more basic

encryption methods produce a lot of post-encryption pattern, the proposed method is able to overcome them. Different from classical methods known to mitigate post-encryption pattern, such as cipher block chaining, one-time pad, and code-based encryption, the proposed method neither employs random generators to produce initialization vectors nor uses large encryption keys resulting in storage difficulties. The presented preliminary results assert the proposed abilities of the method. However, more sophisticated tests are still necessary to assert whether the method is cryptographically secure or not.

Acknowledgement

This research, according for in Article 48 of Degree no 6.008/2006, was funded by Samsung Electronics of Amazonia Ltda, under the terms of Federal Law no 8.387/1991, through agreement no 004, signed with CETELI/UFAM. This work is also supported by FAPEAM, under notice/resolution/settling 003/2019 – POSGRAD 2019.

References

Barker, E. B., Barker, W. C., Burr, W. E., Polk, W. T., and Smid, M. E. (2007). Sp 800-57. Recommendation for key management, Part 1: General (Revised). Technical report, Gaithersburg, MD, United States.

Bellovin, S. M. (2011). Frank miller: Inventor of the one-time pad. *Cryptologia*, 35(3):203–222. DOI: 10.1080/01611194.2011.583711.

Dierks, T. and Rescorla, E. (2006). The transport layer security (TLS) protocol version 1.1. *RFC*, 4346:1–87. Network Working Group, RFC4346.

Ehrsam, W. F., Meyer, C. H., Smith, J. L., and Tuchman, W. L. (1978). Message verification and transmission error detection by block chaining. US Patent 4,074,066.

FIPS (1977). *Data encryption standard (DES)*. Federal Information Processing Standards. Publication 46-3. FIPS 46.

FIPS (1981). *Des modes of operation*. Federal Information Processing Standards. Publication 81. FIPS 81.

NSA (2009). *The Case for Elliptic Curve Cryptography*. U.S. National Security Agency. Archived from the original on 2009-01-17.

NSA (2016). *Commercial National Security Algorithm Suite and Quantum Computing FAQ*. U.S. National Security Agency.

Rijmeanants, D. (2012). The complete guide to secure communications with the one time pad cipher. *Cipher Machines and Cryptology*. Available at `http://users.telenet.be/d.rijmenants`.

Sendrier, N. (2017). Code-based cryptography: State of the art and perspectives. *IEEE Security and Privacy*, 15(4):44–50. DOI: 10.1109/MSP.2017.3151345.

Tao, C., Diene, A., Tang, S., and Ding, J. (2013). Simple matrix scheme for encryption. In Gaborit, P., editor, *Post-Quantum Cryptography*, pages 231–242, Berlin, Heidelberg. Springer Berlin Heidelberg.