Journal of Applied Mathematics & Bioinformatics, vol.9, no.1, 2019, 1-13
ISSN: 1792-6602 (print), 1792-6939 (online)
Scientific Press International Limited

Modification and customization of cryptographic algorithms

George Marinakis[1]

Abstract

At the beginning of this study, we examine the different methods of cryptographic algorithm modifications and we separate them into two major categories, the "internal" and "external" modification. After this, we focus on external algorithm modification and we propose a multiple encryption scheme for the weigh between the performance and the security of the algorithm. And finally, we introduce the concept of dynamic algorithm cascading with which we can increase the security of multiple encryption.

Mathematics Subject Classification: cryptographic algorithm, cryptanalytic attacks.
Keywords: 94A60

1 Introduction

Cryptographic algorithms must be continuously improved, to encounter the various cryptanalytic attacks, which are evolving into two major areas:

a) Faster exhaustive search of the key (Brute Force Attack), due to the increase of computer power.

b) Ongoing detection and exploitation of the vulnerabilities of the internal algorithm structure, as long as its life time is extending.

Due to the above reasons, every new cryptographic algorithm must include in its design counter measures for all the known vulnerabilities and cryptanalytic attacks of the older algorithms. Therefore, a new algorithm is not a totally new

[1] gmari@tee.gr

Article Info: *Received*: May 29, 2019. *Revised*: June 20, 2019
Published online: June 30, 2019

construction, but it includes major and minor modifications of older algorithms, in order to increase its immunity to cryptanalytic attacks. Besides that, the modification of an existing algorithm instead of designing a new one, is an attractive solution because it reduces significantly the necessary research and development cost. For the purpose of this study, we will range the cryptographic algorithm modifications into two basic categories:

- Internal modification
- External modification

2 Internal modification

With the term "internal modification" we mean the change of a basic structural unit of the algorithm architecture, in order to increase its non linear behavior and its complexity. This modification may offer a maximum cryptographic strength, but it needs a great experience in order not to reduce or even destroy the initial algorithm security. Aside from this, the time and cost for the research/development of the algorithm modification and for its implementation in hardware/software, can be extremely high.

One general example of internal modification, could be the increase of the key length of the algorithm or the addition of a secondary key. In a block cipher, the modification could be the change of its S-boxes or the increase of the number of rounds (permutations). And in a stream cipher, we can change the number and the structure of the LFSR or change their feedback paths.

In Table 1 we give a historical survey, where the left column shows some of the most known cryptographic algorithms and the right column shows the original algorithms from which they came from after modification.

Table 1: Modifications of some known cryptographic algorithms

ALGORITHM MODIFICATIONS **A brief history**		
DES (1976)	based on	**Lucifer** (1971)
DES-X (1984)	based on	**DES** (1976)
IDEA (1991)	based on	**PES** (1990)
MMB (1993)	based on	**IDEA** (1991)
Triple DES (1995)	based on	**DES** (1976)
ICE (1997)	based on	**DES** (1976)
AES (1998)	based on	**Square** (1997)

Anubis (2000)	based on	**AES** (1998)
Grand Cru (2000)	based on	**AES** (1998)
MESH (2002)	based on	**IDEA** (1991)
IDEA NXT (2003)	based on	**IDEA** (1991)

Some manufacturers offer to the users the capability to modify their algorithm internal structure, a process which is called customization or programmable cryptography. It is obvious that the customization must be carefully implemented by the user under a thorough training by the manufacturer, who must provide and the necessary hardware/software tools for the design and testing of the new algorithm, in order not to affect the security of the initial algorithm. As long as the customization remains secret, it can give an additional security to the user. However, the key issues are the deep know-how and the confidentiality of the manufacturer, otherwise either the security of the initial algorithm may be decreased, either the customized algorithm may be compromised.

We must point out that in every case of “internal modification” the designer must try to satisfy the following requirements :

- Conduct many different randomness tests, in as many as possible output samples of the new algorithm.
- Select the proper hardware/software implementation for the optimum physical security, performance and flexibility.
- Maintain a strict configuration management of all the major and minor modifications through the lifetime of the algorithm.
- Include the capability of a variable key length, for the weighing between the performance and the security of the algorithm.

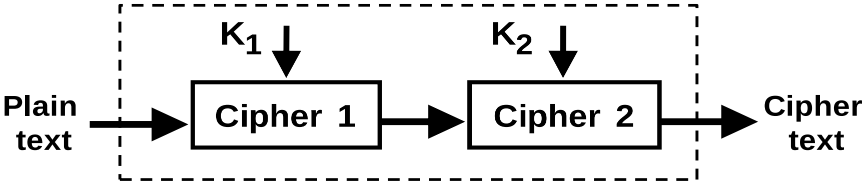
3 External modification

With the term “external modification” we mean the addition of external units to an algorithm, without changing its internal structure. The total result is a new “hybrid” algorithm, which includes the original algorithm as a core. The external modification is a more safe and convenient method, if the designer does not have a high experience in cryptographic algorithm design. Furthermore, it offers significant design flexibility and reduces the research and development time and cost.

In the following paragraphs we will describe the most important methods of external modification which are : Multiple encryption (Cascading), Pre and Post processing, Pre and Post whitening and Tweaking.

3.1. Multiple encryption (Cascading)

The method of multiple encryption (cipher cascading) can increase the complexity of a cryptographic system, especially if the algorithms are different and their keys are different and independently chosen. In Figure 1 we give the general concept of multiple encryption. Obviously, at the decryption process of multiple encryption, the order of the algorithms and the keys is reversed.



$$C = E_2 (E_1 (P , K_1) , K_2)$$

$$\text{Total key length } k = k_1 + k_2 \qquad \text{Entropy } H = k_1 + k_2$$

Figure 1: Double encryption (two stage cascading)

In Figure 2 we see the multiple encryption examples of Double DES and Triple DES. We see that in the case of Double DES, due to the meet in the middle attack, the active key length is 56 bits instead the 112 bits which is the theoretical value **[3], [4]**.

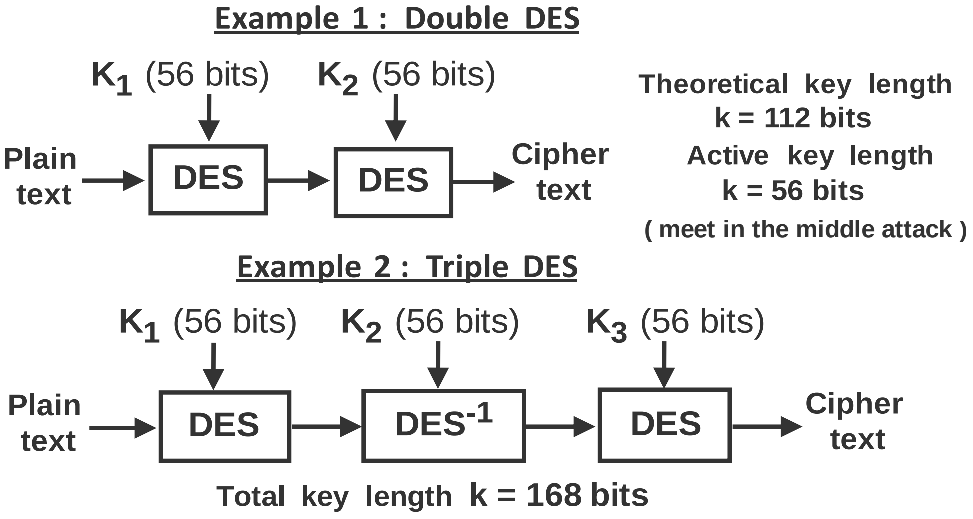


Figure 2: Double DES and Triple DES

We must note that one disadvantage of the multiple encryption systems is the latency which is generated in every cascade stage.

3.2. Pre and Post processing

As is shown in Figure 3, with Pre and Post processing the plain data are transformed before the encryption and at the receiver's site they are retransformed with the reverse process after the decryption. The transformation could be a bit permutation or a bit compression method. The complexity of the Pre and Post processing is weaker than the multiple encryption as long as the transformation process does not use any key.

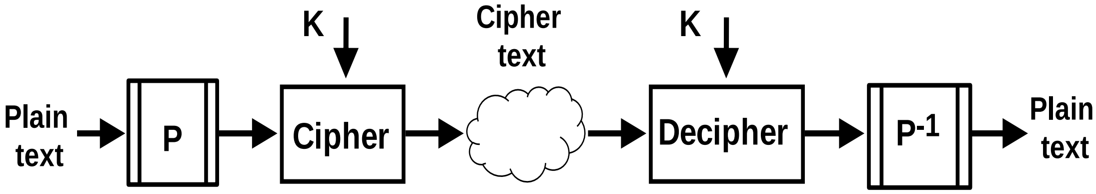


Figure 3: Pre and Post processing (where P is a bit permutation or a bit compression)

3.3. Pre and Post whitening

In the Pre and Post whitening, we use two extra keys, where the first is XORed with the data before the encryption and the second is XORed with the data after the encryption. As is shown in the example of Figure 4, the bits of K_1 are XORed with the plain blocks before they enter the AES cipher and then the bits of the K_2 are XORed with the ciphered blocks of AES. This means that the size of keys K_1 and K_2 must be equal to the size of the blocks. At the decryption process, the order of the XORing is reversed.

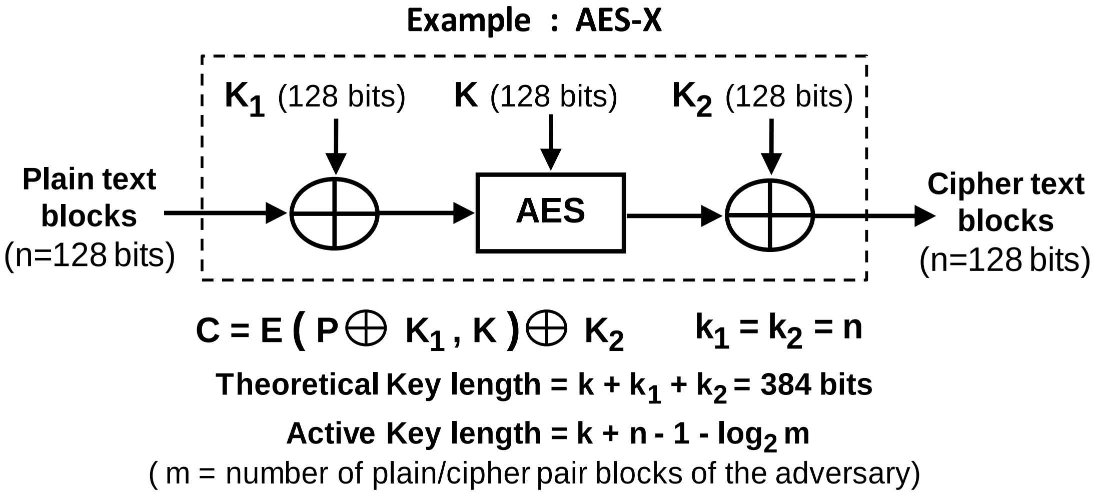

Example : AES-X
K1 (128 bits)
K (128 bits)
K2 (128 bits)
Plain text blocks (n=128 bits)
AES
Cipher text blocks (n=128 bits)
C = E (P ⊕ K1 , K) ⊕ K2
k1 = k2 = n
Theoretical Key length = k + k1 + k2 = 384 bits
Active Key length = k + n - 1 - log2 m
(m = number of plain/cipher pair blocks of the adversary)

Figure 4: Example of Pre and Post whitening : AES-X

3.4. Tweaked block ciphers

In the "Tweaked block ciphers" (Figure 5) the input and output of the cipher are XORed with the output of a special function h(T), which takes as input the value T (Tweak). This makes T to act as an extra key. The sizes of T and the output of h(T) must be equal to the size of the blocks **[5], [6]**.

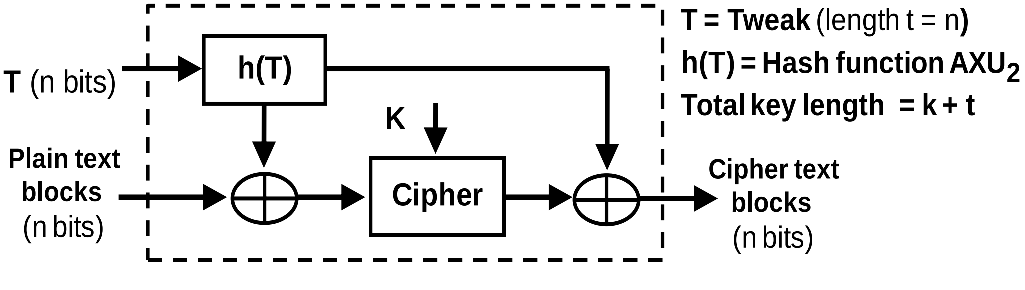

T = Tweak (length t = n)
h(T) = Hash function AXU2
Total key length = k + t
T (n bits)
h(T)
K
Plain text blocks (n bits)
Cipher
Cipher text blocks (n bits)

Figure 5: Tweaked block cipher

In Figure 6, we see a scheme for the encryption of block storage data, which uses two AES ciphers with keys K_1, K_2 and two tweaks **m** and **n**, which correspond to the numbers of the storage sector and the storage block **[7].**

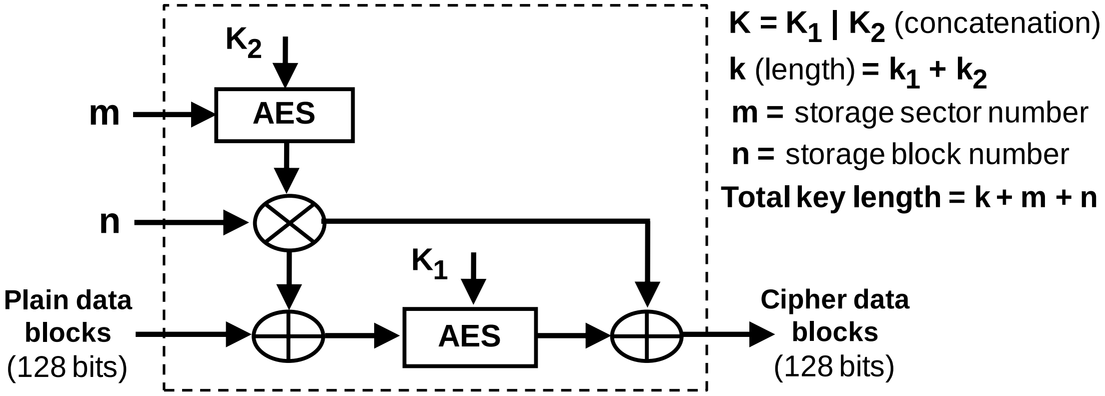


Figure 6: Tweaked block cipher example : AES-XTS (encryption of storage data)

4 Weighing security and performance with multiple encryption

As we noted, the two basic internal modifications of cryptographic algorithms, are the increments of their internal complexity and their key length. However, these two increments can decrease the performance (speed), therefore sometimes is required a weighing between the performance and the security of the algorithm. This means that for a big amount of low classified data, we need to use less complex algorithms with smaller key lengths in order to save time. This can be done with the encryption scheme of Figure 7, with the use two selectable algorithms, where A is stronger than B (grater complexity and/or grater key). Here, we can apply the concept of Suite A and Suite B algorithms used by NSA (similar to NATO and E.U. policy), where A is a classified algorithm and B is a published algorithm **[8]**, **[9]**, **[10]**.

As it is shown in Figure 7, after the security separation o f the data, the low classification data are encrypted only with Type B algorithm (key K_B), the medium classification data are encrypted only with Type A algorithm (key K_A), and the high classification data are encrypted with Type A plus Type B algorithm (key K_A+K_B). We must make two notes:

a). According to **[11]**, in a multiple encryption system it is important to put the strongest algorithm in the first position of the cascade.

b). If Type A algorithm is classified, the meet in the middle attack can not be applied.

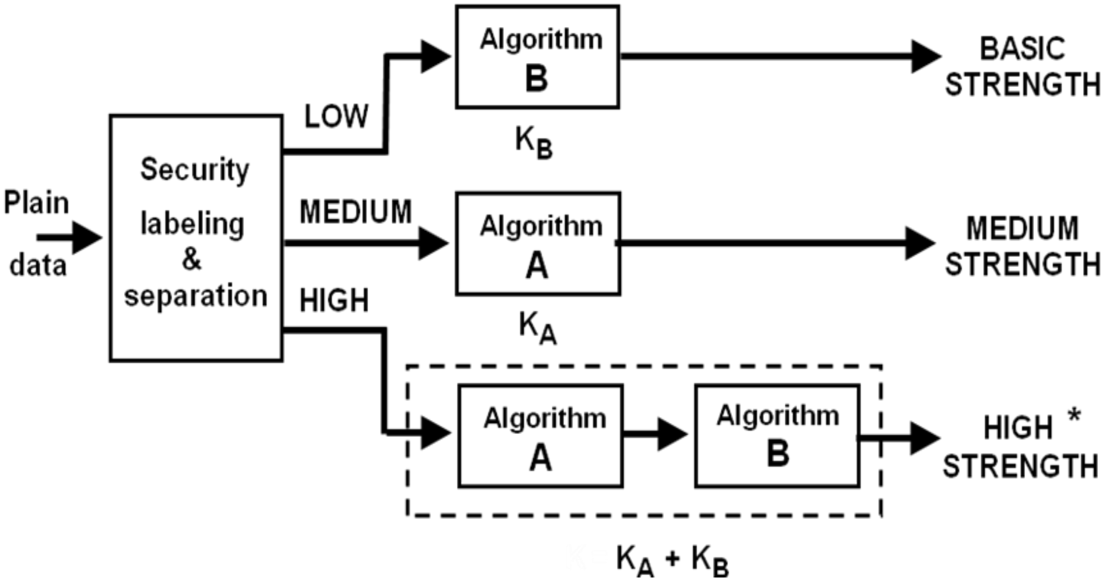


Figure 7: Example of weighing security and performance with double encryption

* It is important to put the strongest algorithm in the first position of the cascade.

* The meet in the middle attack can not be applied, if Type A algorithm is classified.

5 Dynamic Cascading of algorithms

We can increase the security of multiple encryption systems with the **dynamic cascading** concept which we introduce in this study. With the term **dynamic cascading** we mean that instead of keeping the cascade order of the algorithms fixed, we can change it every time that a new key is loaded.

Let's suppose that we have 4 available algorithms A, B, C, D in order to construct a cascade pair (Figure 8). This means that there are 4^2 =16 possible algorithm pairs (AA, AB, AC, AD, BA, BB, BC, BD, CA, CB, CC, CD, DA, DB, DC, DD). When a new key is loaded, we want to select one of these cascade pairs.

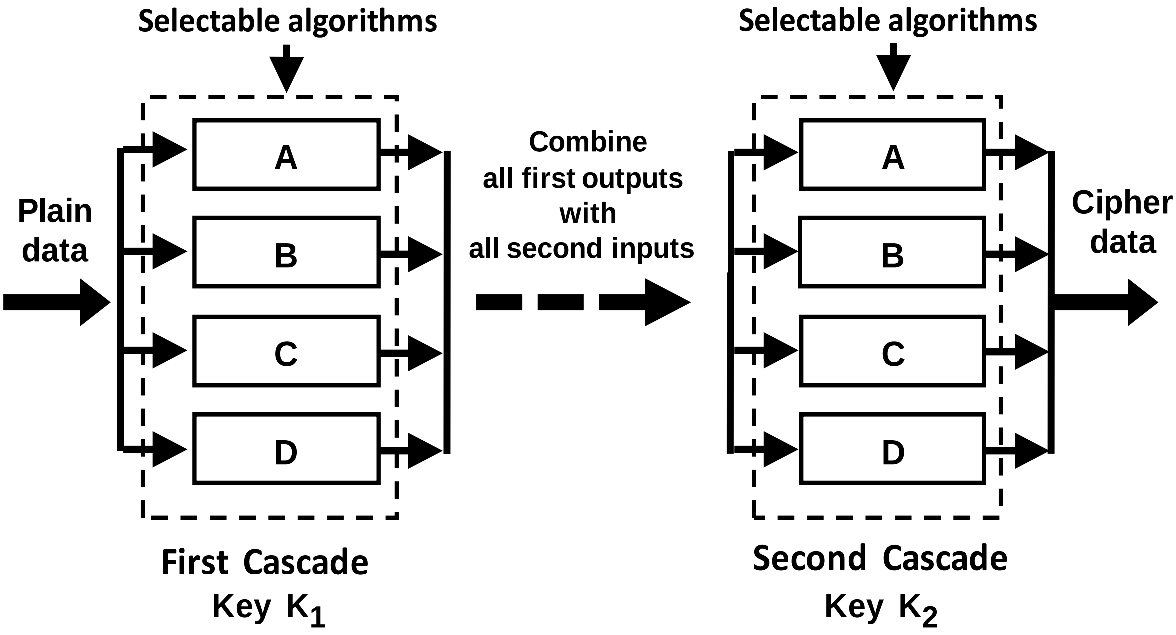


Figure 8: Two stage dynamic cascading, using four selectable algorithms

* The order of the cascaded algorithms changes with every new key pair k1, k2
* The algorithms must have compatible outputs / inputs
* The keys of each cascade stage must be independently chosen

Cascade pair selection procedure

In order to determine one of the above 16 pair combinations, we can use 4 bits of the new key. The selection procedure of the algorithm cascade pair must be done automatically in software and we describe it into two steps:

Step 1 : We predefine 4 bit positions a, b, c, d inside the space of key K_1. In the example of Figure 9, from the total 128 bits of the key K_1 , we have defined a= the 112 bit of the key , b= the 103 bit of the key , c= the 68 bit of the key and d= the 21 bit of the key.

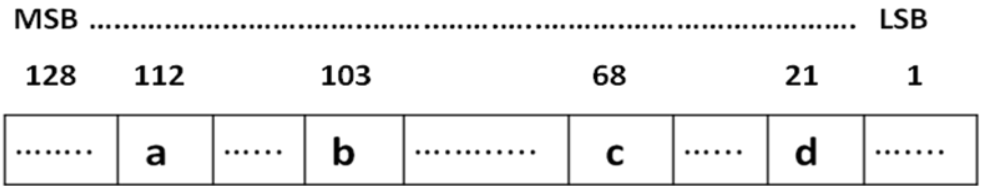

MSB .... LSB
128
112
103
68
21
1
....
a
......
b
....
c
......
d
....

Figure 9: Definition of bits a, b, c, d from the bits of the Key (example)

Step 2 : After each different key set up, first we find the bit values of a, b, c, d and then we put them in Table 2, in order to determine which will be the corresponding cascade pair.

16 ALGORITHM PAIRS							
a b c d		**a b c d**		**a b c d**		**a b c d**	
0001	**AA**	**0101**	**BA**	**1001**	**CA**	**1101**	**DA**
0010	**AB**	**0110**	**BB**	**1010**	**CB**	**1110**	**DB**
0011	**AC**	**0111**	**BC**	**1011**	**CC**	**1111**	**DC**
0100	**AD**	**1000**	**BD**	**1100**	**CD**	**0000**	**DD**

Table 2: The sixteen potential cascade pairs of the four algorithms A, B, C, D.

Dynamic cascading complexity

The benefit of dynamic cascading is that it increases the complexity of the cascade pair for an exhaustive key search attack (under the precondition that the adversary does not know the selection method of the algorithm pair). In practice, it is like adding to an algorithm a secondary key.

As we saw in paragraph 3.1(multiple encryption), for a two stage static cascade, the total complexity for an exhaustive key search attack is :

$$\textbf{Static cascade complexity} = 2^{k1} \ast 2^{k2} = 2^{k1 + k2}$$

where k_1 and k_2 , are the key lengths of first and second cascaded algorithm. Therefore, the entropy of a two stage static cascade system is:

$$\textbf{Entropy } H_S = k_1 + k_2$$

For a two stage dynamic cascade pair with the use of 4 algorithms, there are $4^2 = 2^4 = 16$ possible combinations. Therefore, the total complexity for an exhaustive key search attack is :

$$\textbf{Dynamic cascade complexity} = 2^{k1} \ast 2^{k2} \ast 2^4 = 2^{k1 + k2 + 4}$$

Therefore, the entropy of a two stage dynamic cascading of four algorithms is:

$$\textbf{Entropy } H_D = k_1 + k_2 + 4$$

The above analysis show that the two stage dynamic cascading with four selectable algorithms, multiplies the "exhaustive key search complexity" by the number of the possible pair combinations which is $4^2 = 16$. If we had five selectable algorithms then the number would be $5^2 = 25$.

Extension to n algorithms and m cascades

In the example of Figure 8, we used two cascade stages with four selectable algorithms. We can extend this example to the use of **m** cascade stages with **n** selectable algorithms, as it is shown in Figure 10.

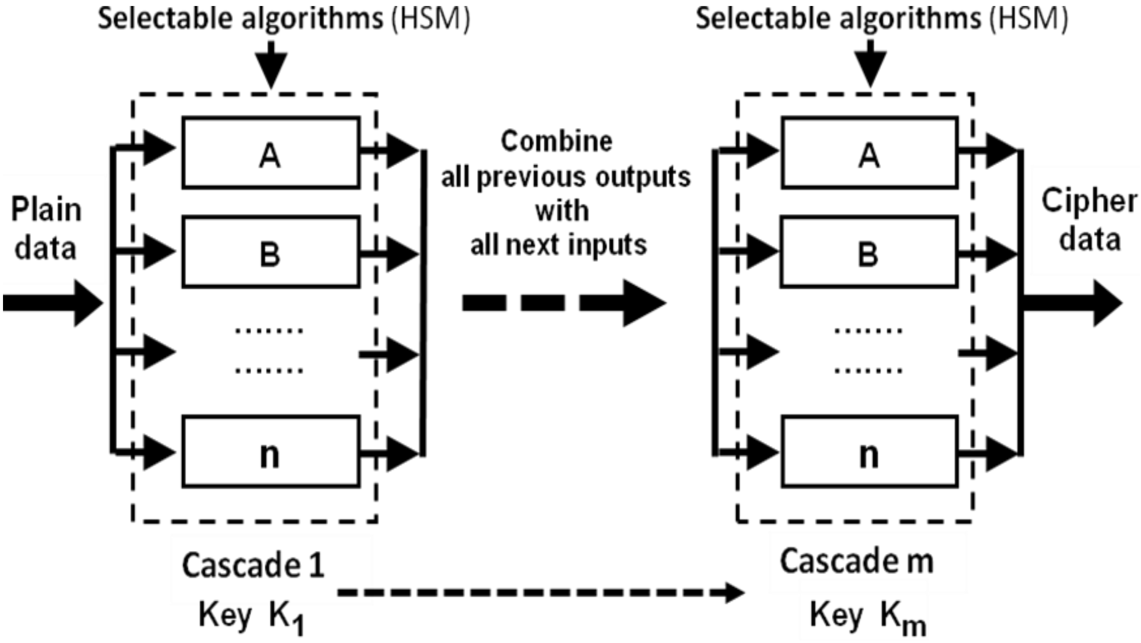


Figure 10: Dynamic cascading with m stages, using n selectable algorithms

In the general case of **m** stage dynamic cascading with the use of **n** algorithms, there are $\mathbf{n^m}$ possible algorithm cascade combinations. Therefore, the total complexity for an exhaustive key search attack is :

$$\textbf{Dynamic cascade complexity} = \mathbf{2^{k1 + k2 + \dots + km} \ast n^m}$$

(where k_1 , k_2 ,…. k_m , are the key lengths of the m cascaded algorithms)
Therefore, the entropy of a n stage dynamic cascading of m algorithms is:

$$\textbf{Entropy } \mathbf{H_D = k_1 + k_2 + \dots\dots k_m + n^m}$$

Example : If we use 3 cascades with 4 algorithms, then the complexity is multiplied by $4^3 = 2^6 = 64$.

The general conclusion from all the above, is that the dynamic cascading multiplies the “exhaustive key search complexity” by the number of the possible cascade combinations.

Concerning the final notes a) and b) of paragraph 4, we believe that the setting of the strongest algorithm in the first position and the risk of the meet in the middle attack are no longer so important, because the adversary needs much more computer power and memory for his attacks, due to the dynamic change of the cascaded algorithms.

Performance of dynamic cascading

As we mention in paragraph 3.1, one disadvantage of the multiple encryption systems is the latency which is generated in every cascade stage. But if the cascaded algorithms are implemented with synchronous hardware programmable units (FPGA, ASIC etc.), this latency can be significantly reduced. One good solution is to implement the selectable algorithms of Figures 8 and 10 with a Hardware Security Module (HSM), which will not only improve the performance (speed) of the cascade system, but it will also save space and power. Many HSMs that are available in the market, integrate up to eight symmetric cryptographic algorithms, various asymmetric cryptographic algorithms and hash functions, plus a Random Number Generator (RNG) for key generation **[12]**, **[13]**, **[14]**, **[15]**.

6 Conclusions

Summarizing all that was presented in this study, we give the following brief conclusions :

a. Internal modifications

The design of “internal” algorithm modifications, needs a deep cryptographic experience and must fulfill the following requirements:

-Include counter measures for all the known cryptanalytic attacks.

-Increase the key length, to avoid Brute Force Attacks due to the continuous evolution of computer power.

-Include a variable key length, for weighing between the performance and the security of the algorithm.

-Conduct many different randomness tests, in as many as possible output samples of the new algorithm.

-Select the optimum implementation for physical security, performance and flexibility. *(Hardware implementation offers better speed, tamper protection and isolation from the network).*

-Maintain a strict configuration management of all the major and minor modifications during the life cycle of the algorithm.

b. External modifications

The “external” algorithm modifications offer better convenience and flexibility and have the following functional and security properties :

-Pre and Post whitening is simpler to design than multiple encryption and offers better total performance.

-Multiple encryption (algorithm cascading) can give greater security, but has lower performance due to the latency of the cascade stages .

-Multiple encryption with only two cascade stages is vulnerable to the “meet in the middle” attack. This can be avoided if at least one of the algorithms is classified.

-A combination of Type A and Type B algorithms can be used in cascade for weighing between security and performance.

-The performance of multi-cascaded algorithm systems can be increased, if they are implemented using Hardware Security Modules (HSM).

-The concept of dynamic algorithm cascading can significantly increase the protection against the exhaustive key search.

References

[1] Alfred Menezes, Paul C. van Oorschot, Scott A.Vanstone, “Handbook of Applied Cryptography”, CRC Press 1997.

[2] Bruce Schneier, “Applied Cryptography”, John Wiley, New York, 1996.

[3] R.C. Merkle and M.E. Hellman"On the Security of Multiple Encryption", Communications of the ACM, July 1981.

[4] NIST.SP.800-57pt1r4 - KEY MANAGEMENT RECOMMENDATIONS-1

[5] M. Liskov, R. Rivest, and D. Wagner “Tweakable block ciphers” In Advances in Cryptology –CRYPTO '02. Lecture Notes in Computer Science, vol 2442, Springer-Verlag, 2002.

[6] P. Rogaway “Efficient Instantiations of Tweakable Blockciphers and Refinements to Modes OCB and PMAC”. Advances in Cryptology - Asiacrypt 2004. Lecture Notes in Computer Science, vol.3329, Springer-Verlag, 2004.

[7] IEEE Std 1619-2007, The XTS-AES Tweakable Block Cipher, Institute of Electrical and Electronics Engineers, Inc., Apr. 18, 2008.

[8] Bob Wheeler “Suite B: Classified Network Security Goes Commercial” , July 2009, The Linley Group, Inc.

[9] “Fact Sheet NSA Suite B Cryptography” https://web.archive.org/web/20051211150701/http://www.nsa.gov/ia/industry/crypto_suite_b.cfm

[10] https://www.zsis.hr/UserDocsImages/Sigurnost/pdfs/TCE621_B_C_AES&DUAL.pdf

[11] Ueli M. Maurer, James L. Massey “Cascade Ciphers : The Importance of Being First” Journal of Cryptology, vol. 6, no. 1, pp. 55-61, 1993.

[12] https://www.thalesesecurity.com/products/general-purpose-hsms

[13] https://safenet.gemalto.com/data-encryption/hardware-security-modules-hsms/

[14] http://realsec.com/en/products-and-solutions/general-purpose-hsms/

[15] https://www.futurex.com/products/category/hardware-security-modules-hsm