All-Or-Nothing Transforms Using Quasigroups

Stelios I. Marnas, Lefteris Angelis, and George L. Bleris

Department of Informatics, Aristotle University
54124 Thessaloniki, Greece
Email: {marnas,lef,bleris}@csd.auth.gr

Abstract. In this paper we suggest a new transformation scheme for All-Or-Nothing encryption, originally suggested by Rivest. The new transform concerns the use of quasigroups for the preprocessing of the data before any ordinary encryption method. We describe a method of constructing random quasigroups and we propose a way of using the advantages of quasigroup in Rivest's method. This combination makes the method faster and maintains the advantages against brute-force attacks.

1 Introduction

Brute-force is the most common attack to any cryptosystem. This kind of attack is passive and the objective is to try every possible key until the plaintext makes sense. Any encryption scheme vulnerable to a ciphertext-only attack is considered to be completely insecure [4]. Rivest in [1] introduced *All-Or-Nothing (AON)* encryption mode in order to devise means to make brute-force search more difficult, by appropriately pre-processing a message before encrypting it. The method is general, but it was initially discussed for block-cipher encryption, using fixed-length blocks. It is an unkeyed transformation, mapping a sequence of input blocks $(x_1, x_2, \ldots, x_s)$ to a sequence of output blocks $(y_1, y_2, \ldots, y_{s'})$ having the following properties:

- Having all blocks $(y_1, y_2, \ldots, y_{s'})$ it is easy to compute $(x_1, x_2, \ldots, x_s)$
- If any output block y_i is missing, then it is computationally infeasible to obtain any information about any input block x_i.

Several transformation methods have been proposed in the literature for AON. In this paper we propose a special transform which is based on the use of a quasigroup. The main idea is to preserve a small-length key (e.g. 64-bit) for the main encryption that can be handled by special hardware with not enough processing power or memory. This gives the method a strong advantage, since we can have strong encryption for devices that have minimum performance.

The organization of this paper is as follows: In section 2 we present the basic principles of AON transforms, in Section 3 we give the definition of quasigroups and discuss their application to encryption, in Section 4 we describe our suggestion for using quasigroups as an AON method and finally in Section 5 we conclude with a discussion on the proposed method.

2 All-Or-Nothing Transforms

The method of AON originally proposed by Rivest [1] is very simple and is based on the need for keeping small-length keys for devices with minimal storage and process capability, but increasing their security as if they were using biggest-length keys. The add-on component of the standard encryption methods is a pre-process stage, during which All-Or-Nothing transformations are being calculated over the data. For the description of the method, we assume that an adversary can obtain one block of the message and decrypt it. The chipher-block chaining (CBC mode) is used for the illustration of the method.

Definition 1. Suppose that a block cipher encryption mode transforms a sequence $(m_1, m_2, \ldots, m_s)$ of s message blocks into a sequence $(c_1, c_2, \ldots, c_t)$ of t ciphertext blocks for $t \geq s$. We say that the encryption mode is *strongly non-separable* if it is infeasible to determine even one message block m_i (or any property of a particular message block m_i) without decrypting all t ciphertext blocks. Rivest in [1], proposed a strongly non-separable mode as follows:

- Transform the message sequence $(m_1, m_2, \ldots, m_s)$ into a pseudo-message sequence $(m'_1, m'_2, \ldots, m'_{s'})$ (for $s' \geq s$) with an AON transform.
- Encrypt the pseudo-message with an ordinary encryption mode with the given cryptographic key K to obtain the ciphertext $(c_1, c_2, \ldots, c_t)$.

Definition 2. A transformation T mapping a message sequence $(m_1, m_2, \ldots, m_s)$ into a pseudo-message sequence $(m'_1, m'_2, \ldots, m'_{s'})$ is called an *All-Or-Nothing Transform (AONT)* if

- The transformation T is reversible: given the pseudo-message sequence, one can obtain the original message sequence.
- Both the transformation T and its inverse are efficiently computable (that is, computable in polynomial time).
- It is computationally infeasible to compute any function of any message block if any one of the pseudo-message blocks is unknown.

An AON transform is strongly non-separable and cannot be considered as an encryption method itself, because it does not use any secret key. It is just a preprocess step which amplifies the actual encryption operation that follows. So this step exists only for converting a message into a pseudo-message and backwards. The original AON transforms, introduced in 1, were called *package transforms* and can be described as follows:

1. Let the input message be $(m_1, m_2, \ldots, m_s)$ (where m_i is a b-bit string)
2. Choose at random a key K for the package transform block cipher.
3. Compute the output sequence $(m'_1, m'_2, \ldots, m'_{s'})$, where $s' = s + 1$ as follows:

$$m'_i = m_i \oplus E_K(i) \text{ for } i = 1,2,\ldots,s .$$

($E_K(.)$ is a b-bit block cipher with the key K and $\oplus$ is the bitwise XOR operation).
Let

$$m'_{s'} = K \oplus h_1 \oplus \ldots \oplus h_s$$

where

$h_i = E_{K_0}(m'_i \oplus i)$ for $i = 1,2,\ldots,s$ and K_0 is a fixed, publicly-known key.

The block cipher for the package transform does not use a secret key and does not have to be the same as the block cipher for encrypting the pseudo-message. The key space for the package transform is assumed to be large enough, so it would be computationally infeasible for someone to obtain it with a brute-force attack. It is obvious that the package transform is invertible:

$$K = m'_{s'} \oplus h_1 \oplus \ldots \oplus h_s$$
$$m_i = m'_i \oplus E_K(i) \text{ for } i = 1,2,\ldots,s$$

If any block of pseudo-message sequence is unknown, the key K cannot be computed and therefore it is computationally infeasible to retrieve any message block.

Many AON transforms have been proposed in the literature. Stinson in [5] introduced a different definition for these transforms. His approach has to do with unconditionally secure transforms, as compared to the conditionally secure schemes considered in [1]. Desai in [6] proved that the method is strong and secure in the Shannon Model of block cipher. He gave a new characterization of AON transforms and a new notion concerned with the privacy of keys that provably captures an exhaustive key-search resistance property. AON property was combined also with hash functions in [7]. In [8] a new mode was suggested and practically the adversary does not know the ciphertext as AON transforms are used after encryption, resulting in the shuffling of the ciphertext. For a thorough review of various AON methods we refer to [10].

3 Quasigroups and encryption

A quasigroup is a groupoid (Q, f) satisfying the law

$$(\forall u, v \in Q)(\exists! x, y \in Q)\ (f(u,x) = v \,\&\, f(y,u) = v)\ .$$

This implies the cancellation laws

$$f(x,y) = f(x,z) \Rightarrow y = z\ ,\ f(y,x) = f(z,x) \Rightarrow y = z$$

and that the equations

$$f(\alpha, x) = b\ ,\ f(y, \alpha) = b$$

have unique solutions x, y for each $\alpha, b \in Q$.

Definition 3 A $k \times n$ Latin rectangle on an alphabet $Q = \{q_1,...,q_n\}$ is an array with entries $q_{ij} \in Q, i = 1,2,...,k \;\&\; j = 1,2,...,n$ such that each row and each column consists of different elements of Q. If $k = n$, then the Latin rectangle is called a Latin Square of order n.

It is obvious that if $Q = \{q_1,...,q_n\}$ is a carrier of a quasigroup $(Q,*)$ then it can be considered as a $n \times n$ Latin Square. Thus the construction of a Latin Square essentially provides a method to obtain a random quasigroup. One such method is given by Hall's theorem [2], which states that any $k \times n$ Latin rectangle can be extended to a $(k+1) \times n$ Latin rectangle, for each $k = 1,2,...,n-1$, and the extension can be made in at least $(n-k)!$ ways. This shows that we can have at least $n!(n-1)!...2!1!n!$ Latin Squares of order n over an alphabet with cardinality n.

Using a quasigroup $(Q,*)$ we define a binary operation $/$ on Q with the following characteristic:

$$x / y = z \Leftrightarrow x * z = y,$$

for all $x, y \in Q$. It is clear that the qroupoid $(Q,/)$ is also a quasigroup.

It is also obvious that the operation $/$ is dual to $*$, and that $(Q,/)$ is a dual quasigroup to $(Q,*)$. Furthermore the algebra $(Q,*,/)$ is a quasigroup, an expansion of $(Q,*)$. For the quasigroup $(Q,*,/)$ it follows that:

$$x/(x * y) = y, \quad x * (x / y) = y$$

With the method we just described, we first construct a Latin Square and we obtain our quasigroup by considering it as the multiplication table of the quasigroup. For the purposes of our AON transform, we employ another method in order to construct very quickly random Latin Squares from which we will create a pair of quasigroups with dual operations. The method gives Latin Squares of order $n = p - 1$, where p is a prime.

In our application, we use a Latin Square of order 256, with elements the numbers $1,...,256$ (note that the number 257 is a prime). The reasons for this choice will be explained below. Our Latin Square is constructed by the following general process:

Step 1. The first row $(a_{11}, a_{12},..., a_{1n})$ is created randomly as a random permutation of the elements $1,...,n$.

Step 2. Every element of the i-th row, $i = 2,...,n$ is calculated by $a_{ij} = i * a_{1j} \bmod p$, where $i, j = 1,2,...,n$ and $p = n+1$ prime.

It is easy to show that the array constructed by the above method is a Latin Square, since for every prime p the set $Z_p = \{0,1,2,..., p-1\}$, with the operations of addition and multiplication $\bmod p$, is a Galois Field *GF(p)*.Indeed, first note that the multiplication $\bmod p$ of any two non-zero elements of Z_p never gives 0. Now, let $a_{ij} = a_{ik}$.

Then, $i * a_{1j} = i * a_{1k}$ and from the algebraic properties of Z_p we have that $a_{1j} = a_{1k}$. Similarly, if $a_{ij} = a_{sj}$, then $i * a_{1j} = s * a_{1j}$ and $i = s$. Thus, it is impossible to have in the same row or column the same element and this means that the constructed array is a Latin Square of order $n = p - 1$.

Example 1. As an illustration, we give a Latin Square of order 6, with $p = 6 + 1 = 7$ constructed by the above method.

	1	2	3	4	5	6
1	3	5	2	1	6	4
2	6(=2*3mod7)	3(=2*5mod7)	4(=2*2mod7)	2(=2*1mod7)	5(=2*6mod7)	1(=2*4mod7)
3	2(=3*3mod7)	1(=3*5mod7)	6(=3*2mod7)	3(=3*1mod7)	4(=3*6mod7)	5(=3*4mod7)
4	5(=4*3mod7)	6(=4*5mod7)	1(=4*2mod7)	4(=4*1mod7)	3(=4*6mod7)	2(=4*4mod7)
5	1(=5*3mod7)	4(=5*5mod7)	3(=5*2mod7)	5(=5*1mod7)	2(=5*6mod7)	6(=5*4mod7)
6	4(=6*3mod7)	2(=6*5mod7)	5(=6*2mod7)	6(=6*1mod7)	1(=6*6mod7)	3(=6*4mod7)

We have chosen this construction method because it is faster and easier to program it than Hall's. Based on the definitions above, it is obvious that the produced Latin Square defines a quasigroup and we can define the dual operation.

Quasigroups have already been proposed for online communication in [3, 9] as an encryption method combining security and speed. The method is simple since the two parties need only to know the pair of the quasigroups used. Then each character is encrypted and transmitted. It is clear that an initializing stage is needed for establishing the communication, the handshaking for exchanging the initial character and the pair of the quasigroups. The method is claimed to be secure, but the main advantage is that is fast. Security and speed are the main reasons for choosing quasigroups, since each character is encoded by one only. Moreover, one can choose a known pair of quasigroups or create each time a new pair randomly, amplifying this way the security.

Our approach uses the above features to give the AON method speed and increase, in a way, its security. We are not interested in the encryption power and completeness of the quasigroup as in 9, since our goal is to preserve main advantages of the two methods and combine them in the best way.

4 The All-or-Nothing Transform With Quasigroups

The idea is based on the AON property and is actually a modification of it. Since the basic disadvantage of AONTs seems to be time, it became a good motive for us to look for a way to reduce time needed. It is obvious that the simplicity of the algorithm doesn't let enough room for improvement, so we preserved the advantage of the all-or-nothing method and replaced only the algorithm that produces the pseudo-message. As discussed in the previous section our method is a combination of AON mode and encryption with quasigroups. So instead of the bitwise XOR used as

basic operation in AONTs, we have used the encryption with quasigroups method as follows:

a) The first row of the quasigroup is randomly created. Then the rest 255 rows are being created with the procedure of Section 3.
b) One of the quasigroups elements, called *leader* (say a_1), required for the initial step of the process, is randomly chosen.
c) With the help of the leader and the quasigroup the pseudo-message is created. Each message block (size of 8 bits) with the help of the quasigroup is mapped into the appropriate output block. The method is the one described in 3 with a few modifications:
   1. We have a pair of quasigroups of order 256, so we can "encrypt" the 256 ASCII characters. Someone can then consider, blocks of 8 bits as binary strings, as already proposed. This way each block is a binary representation of a number between 0 and 255 and it can be mapped to a unique element ofthe quasigroup. Since the construction method we described uses as elements the numbers 1 to 256, we just have to replace 256 by 0 in order to use the 8-bit representation.
   2. The pseudo-message is created with the following procedure:
      Let $Q = \{q_1, q_2 ..., q_n\}$ $(n \geq 1)$ be an alphabet and let $(Q, *, /)$ be the quasigroup defined above. The two unary operations f_* and $f_/$ are defined as follows:

      Definition 4. Let $u_i \in Q$, $k \geq 1$.
      Then
      $$f_*(u_1 u_2 ... u_k) = v_1 v_2 ... v_k$$
      Where
      $$v_1 = a_1 * u_1,\ v_{i+1} = v_i * u_{i+1},\ i = 1,2,...,k-1,$$
      $$f_/(v_1 v_2 ... v_k) = u_1 u_2 ... u_k$$
      where
      $$u_1 = a_1 / v_1,\ u_{i+1} = v_i / v_{i+1},\ i = 1,2,...,k-1.$$

   3. The pseudo-message is extended with the leader a_1 and the first row of the quasigroup as follows:

      message to encrypt =
      leader a_1 + 1st row of the quasigroup + pseudo-message

Then the actual encryption takes place with any known algorithm. The pseudo-message is a little longer than the original one since it is extended with the 256 elements of the 1st row of the quasigroup and the leader needed for the initial step of the conversion, but this isn't important since the total cost is 257 bytes of added information.

Based on the previous paragraph, the communication between two parties is the following:

SENDER:

- Creates the quasigroup randomly as described in Section 3.
- Converts the message into the pseudo-message with the help of the leader.
- Creates the actual pseudo-message as defined previously, adding the necessary information for obtaining the message.
- Performs the actual encryption using any of the well-known algorithms.
- Transmits the cipher.

RECEIVER:

- Receives the cipher and decrypts it.
- Obtains the pseudo-message, which contains the "key-information" to convert it to the actual message.
- Obtains by the first byte the leader while the rest 256 bytes comprise the first row of the quasigroup. Then creates the pair of quasigroups.
- Converts the rest of the pseudo-message and gets the actual message, with the help of the leader and the dual operation of the quasigroup.

Example 2.
Let m be the original message to be encrypted and transmitted. The sender creates randomly the quasigroup. We will use the one already created in Example 1. For simplicity we will use quasigroup of order 6 and numbers instead of characters. Note that one byte is 8 bits, so we use 8-bit blocks.

*	1	2	3	4	5	6
1	3	5	2	1	6	4
2	6	3	4	2	5	1
3	2	1	6	3	4	5
4	5	6	1	4	3	2
5	1	4	3	5	2	6
6	4	2	5	6	1	3

SENDER:
Let $m = 425461134425633$ be the message for transmission.
We choose the leader $a_1 = 4$ randomly. The conversion of the message is the following:

$$m' = 461145122234241$$

and the pseudo-message is:

$$m'' = 4352164461145122234241$$

The message above is ready for encryption with any known algorithm.

RECEIVER:
The received message is decrypted and the retrieved pseudo-message is:

$$m'' = 4352164461145122234241$$

The receiver obtains the leader $a_1 = 4$ (first byte of the message)
The quasigroup is obtained from the following 8 bytes. With the use of the relation:

$$x/(x*y)=y\,,\quad x*(x/y)=y$$

the receiver obtains the binary operation :

/	1	2	3	4	5	6
1	4	3	1	6	2	5
2	6	4	2	3	5	1
3	2	1	4	5	6	3
4	3	6	5	4	1	2
5	1	5	3	2	4	6
6	5	2	6	1	3	4

Using the leader $a_1 = 4$, the quasigroup above and Definition 4 the message $m = 425461134425633$ it can be easily recovered from $m' = 461145122234241$.

5 Conclusions

In this paper, a bridge was held between the AON encryption mode and encryption with Quasigroups. Our objective was to preserve the advantages of the use of block-ciphers since sometimes it is preferable to gain as much security as is possible, rather than introduce something completely new, as that would automatically make existing cryptographic hardware useless. Our suggestion preserves the advantage of all-or-nothing encryption mode, the pre-processing mode, which is responsible for the increased security. The penalty for that according to [7] is the extra time needed for that stage, which is the main withdraw. Thus, we have thought to use a fast encryption method in order to reduce that time. The encryption with quasigroups is suitable for online communications, so its main characteristic is the converting speed. It seemed to us that even if we didn't take full advantage of the encryption strength of quasigroups, the result of using the method, as a pre-processing stage, would improve all-or-nothing transforms at least in time.

Of course the improvement of time is crucial, as nowadays is very important to spend as less time as possible, but also to preserve an acceptable level of security. For example a transaction with a smart card should be as fast as possible and of course as secure as possible. But since there are limitations in hardware (processing power, memory, storage capability, etc) and the number of daily transactions could be high, it is important to suggest methods that would make everyday life easier, emphasizing on security.

Our method is different from the ones suggested until now. Our future plans include comparisons with some already known methods, in order to get results for the level of security and especially for the processing speed.

References

1. R.L. Rivest, *All-or-nothing Encryption and the Package Transform*. Fast Software Encryption '97, Springer LNCS Vol.1267, (1997)
2. M. Hall, *Combinatorial Theory*, John Wiley, New York, 2nd edition, (1986)
3. S. Markovski, D. Gligoroski, S. Andova, Using Quasigroups for One-one secure E|ncoding. *Proceedings 8th Conference Logic and Computer Science (LIRA),* Novi Sad, (1997) 157-162
4. A.J. Menezes, P.C. van Oorschot, S.A. Vanstone, *Handbook of Applied Cryptography*, CRC Press 1997
5. D. R. Stinson, *Something About all or Nothing (transforms*. Design, Codes and Cryptography) (2001) 133-138
6. *A. Desai,* The Security of All-or-nothing Encryption: Protecting Against Exhaustive Key *Search, Proceedings Crypto Conference,* Springer LNCS Vol.1880, (2000) 359-375
7. S. Shin, K. Hyune Rhee, A New Design of the Hash Functions With All-or-Nothing Property, *Journal of Information Science and Engineering* 17:945-957, (2001)
8. V. Canda, T. van Trung, A New Mode of Using All-Or-Nothing Transforms*,* http:// www.exp-math.uni-essen.de/~trung/papers.html
9. S. Markovski, D. Gligoroski, D. Stojcevska, Secure two-way on-line communication by using quasigroup enciphering with almost public key, Novi Sad Jornal Mathematics, 30(2):43-49, (2000)
10. V. Boyko, On All-or-Nothing Transforms and Password-Authenticated Key Exchange Protocols*,* Ph.D. Thesis Department of Electrical Engineering and Computer Science, MIT (2000)