A New Post-Quantum Key Agreement Protocol and Derived Cryptosystem Based on Rectangular Matrices

HUGO DANIEL SCOLNIK [1, 2, 3, 4] hugo@dc.uba.ar, hscolnik@gmail.com

JUAN PEDRO HECHT [3] phecht@dc.uba.ar, qubit101@gmail.com

[1]*Instituto de Ciencias de la Computación, Universidad de Buenos Aires and CONICET, Buenos Aires, Argentina.*

[2]*Departamento de Computación, Facultad de Ciencias Exactas y Naturales, Universidad de Buenos Aires, Buenos Aires, Argentina.*

[3]*Maestria en Seguridad Informática – Facultades de Ciencias Económicas, Ciencias Exactas y Naturales, Ingeniería, Universidad de Buenos Aires, Buenos Aires, Argentina.*

[4]*Corresponding author*

Abstract. In this paper, we present an original algorithm to generate session keys and a subsequent generalized ElGamal-type cryptosystem. The scheme here presented has been designed to prevent both linear and brute force attacks using on one hand rectangular matrices, and on the other achieving very high complexity. Moreover, analytical attacks are NP-complete. An interesting result of our protocol is that the secret shared key is an invariant obtained from both public and private information using usual multiplications of matrices, either in numerical or $\mathbb{F}_{2^m}$ polynomial fields so that both sorts of operations could be eventually combined to increase even more the security against classical and quantum attacks.

Keywords: key-exchange protocol, non-commutative algebraic cryptography, post-quantum cryptography, rectangular matrices, semantic security.

Statements and declarations: the authors declare no competing financial interests. No funding was received for conducting this study.

1. Introduction

As it is very well known generating secure key exchange algorithms is a priority for implementing asymmetric protocols [16]. The idea of public key cryptography goes back to the work of James Ellis [7] and the seminal work of Diffie-Hellman [6] which was the first practical solution universally used in SSL, TLS, SSH, IPsec, PKI, Signal, etc.
On the other hand, the imminent appearance of quantum computers able to implement Shor's and Grover's algorithms [2] which seriously affect the currently used cryptographic methods, led to the current research efforts in Post Quantum Cryptography (PQC).
This paper was inspired by E. Stickels's proposals [24] which were cryptanalyzed by V. Shpilrain [19] and C. Mullan [18]. More recently, S. Kanwal and R. Ali [11] published an interesting protocol but it was also cryptanalyzed by J. Liu et al. [14]. A natural alternative

was to use rank-deficient matrices but this has been cryptanalyzed by F.Virdia using Jordan canonical forms[26].
It is worthwhile to point out that in the NIST competition for standardization of post-quantum protocols [20], there is none based on the use of non-commutative algebraic systems [19], those dedicated to key exchange protocols (KEP) and their canonical asymmetric cryptosystems, derived using a generalized ElGamal scheme. This paper aims to provide an alternative solution in this regard.

2. The notation used in this work

p: prime integer, Z_p: set of non-negative residuals mod p, products in Z_p (represented by dots), ||: concatenation, Det[A]: the determinant of matrix A, A^T: transpose of matrix A, A(i, j): matrix component of the i-th row and j-th column, $\in_{rand}$: random uniform selection in a closed interval, $\oplus$: bitwise XOR.

3. Paper organization

First, we present an overall description of the proposed algorithms and the corresponding protocols, the proof that Alice and Bob will derive a common key, security considerations, and finally some experimental results and a discussion.

4. Overall description

The algorithm starts by choosing a prime p shared by Alice and Bob who generate two rectangular matrices each, the first one with more rows than columns and the second one with inverse dimensions, and t is the number of iterations. For each iteration and every entry, a random integer $s \in_{rand} [(p-1)/2, p-1]$ is chosen as the module, employing the algorithm given in [5]. The entries of the generated matrices are randomly selected from Z_s.
Following this scheme, Alice calculates two matrices $A1_k$ and $B1_k$ in each cycle (k=1,…,t) and computes U_k utilizing the matrix product

$$U_k = A1_k \,.\, B1_k \ (\text{mod } p) \quad (k=1,2,3,\ldots,t)$$

The vector $U=(U_1, U_2, U_3, \ldots, U_t)$ is sent to Bob. Analogously Bob computes

$$V_k = A2_k \,.\, B2_k \ (\text{mod } p) \quad (k=1,2,3,\ldots,t)$$

and the vector $V=(V_1, V_2, V_3,\ldots, V_t)$ is sent to Alice.

We prove that:

$$\text{A-KEY}_k = \text{Det}[A1_k^T \,.\, V_k.\, B1_k^T] \ (\text{mod } p)$$

and

$$\text{B-KEY}_k = \text{Det}[A2_k^T \,.U_k.\, B2_k^T] \ (\text{mod } p)$$

are equal in each k-cycle.

Finally, Alice computes the hashing of A-CONCAT = A-KEY_1 || A-KEY_2 ||… || A-KEY_t, and Bob the hashing of B-CONCAT = B-KEY_1 || B-KEY_2 ||… || B-KEY_t which are equal, and hence this is the shared key.
We must observe this protocol is highly parameterizable since we can change the dimensions of the matrices, the number of cycles, the primes, etc. The numerical results

(see below) show a very complex shared key can be obtained in a fraction of a second using a standard I7 processor.

5. Key exchange algorithm

ALGORITHM 1: PQC multiKEP

COMMENTS
The key Exchange Algorithm (KEP) uses several internal cycles as defined below, therefore defined here as a multiKEP.
INPUT: see the initial configuration.
OUTPUT: shared session key of 512-bits.
INITIAL CONFIGURATION (PUBLIC VALUES):
p: a shared prime number that can be obtained randomly.
rows[X|, columns[X]: dimensions of the matrices X:{A, B}, where rowsA=columnsB, columnsA=rowsB and rowsA > columnsA.
rowsA is a value whose maximum is a predefined rowmax value. Our proposal is rowmax=100, rowsA $\in_{rand}$ [5, rowmax] and columnsA $\in_{rand}$ [4, rowsA-1].
t: number of iterations
H( ): hashing SHA3-512.

ALICE

```
1.  for k=1 to t
2.      for i=1 to rowsA
3.          for j=1 to columnsA
4.              $A1_k$ (i,j) $\in_{rand}$ [(p-1)/2, p -1]
5.          next j
6.      next i
7.      for i=1 to rowsB
8.          for j=1 to columnsB
9.              $B1_k$ (i,j) $\in_{rand}$ [(p-1)/2, p -1]
10.         next j
11.     next i
12.     $U_k = A1_k . B1_k$ (mod p)
13. next k
14. Send the vector $U = (U_1, \dots , U_t)$ to Bob
```

BOB

```
15. for k=1 to t
16.     for i=1 to rowsA
17.         for j=1 to columnsA
18.             $A2_k$ (i,j) $\in_{rand}$ [(p-1)/2, p -1]
19.         next j
20.     next i
21.     for i=1 to rowsB
```

```
22.           for j=1 to columnsB
23.                 $B2_k(i,j) \in_{rand} [(p-1)/2, p-1]$
24.           next j
25.       next i
26.       $V_k = A2_k . B2_k \pmod p$
27.  next k
28.  Send the vector V = (V1, … ,Vt) to Alice
```

SESSION KEY OBTAINED BY ALICE

```
29.  for k=1 to t
30.       $A\text{-}KEY_k = Det[A1_k^T . V_k . B1_k^T] \pmod p$
31.  next k
32.  $A\text{-}CONCAT = A\text{-}KEY_1 \| A\text{-}KEY_2 \| \dots \| A\text{-}KEY_t$
33.  $KEY_{alice} = H(A\text{-}CONCAT)$
```

SESSION KEY OBTAINED BY BOB

```
34.  for k=1 to t
35.       $B\text{-}KEY_k = Det[A2_k^T . U_k . B2_k^T] \pmod p$
36.  next k
37.  $B\text{-}CONCAT = B\text{-}KEY_1 \| B\text{-}KEY_2 \| \dots \| B\text{-}KEY_t$
38.  $KEY_{bob} = H(B\text{-}CONCAT)$
```

6. Algorithm 1: keys equality proof

Lemma 1:
The keys given by Algorithm 1 are equal, that is $KEY_{alice} = KEY_{bob}$

Proof: it is very simple taking into account the elementary properties $det(X)=det(X^t)$, $det(XY)=det(X)det(Y)$, $(XY)^t = Y^t X^t$ where X, Y are square matrices of the same dimension We have to prove that for every k (all operations (mod p))

Alice's
$KEY_{alice} = Det[A1_k^T . V_k . B1_k^T) = Det[A1_k^T . A2_k . B2_k . B1_k^T) =$
$= Det[(A2_k^T . A1_k)^T . (B1_k . B2_k^T)^T] = Det[(A2_k^T . A1_k)^T] . Det[(B1_k . B2_k^T)^T] =$
$= Det[A2_k^T . A1_k] . Det[B1_k . B2_k^T]$

Bob's
$KEY_{bob} = Det[A2_k^T . U_k . B2_k^T] = Det[A2_k^T . A1_k . B1_k . B2_k^T] =$
$= Det[A2_k^T . A1_k] . Det[B1_k . B2_k^T] = KEY_{alice}$ ■

7. Derived cipher algorithm

ALGORITHM 2: PQC multiKEP + ElGamal cipher

Observation: Vector U was received by Bob

Insert here algorithm 1 (up to line 27.)

BOB CIPHERS A MESSAGE TO ALICE

```
30.  msg = 512-bit message from Bob
31.  for k=1 to t
32.      B-KEY_k = $Det[A2_k^T . U_k . B2_k^T] \pmod p$
33.  next k
34.  B-CONCAT = $B\text{-}KEY_1 \| B\text{-}KEY_2 \| \ldots \| B\text{-}KEY_t$
```

ELGAMAL (C, D):

```
35.  C = V    (* see Algorithm 1 *)
36.  D = H[ B-CONCAT ] $\oplus$ msg
37.  Send (C, D) to Alice
```

ALICE RECOVERS THE MESSAGE FROM BOB

```
38.  for k=1 to t
39.      A-KEY_k = $Det[A1_k^T . C_k . B1_k^T] \pmod p$
40.  next k
41.  A-CONCAT = A-KEY1 || A-KEY2 ||… || A-KEYt
42.  msg =H[ A-CONCAT ] $\oplus$ D
```

The above algorithms can be enhanced by combining modular operations in Z_p with polynomial multiplications in a finite field like $\mathbb{F}_{2^m}$, see Discussion below [12, 4]. A convenient choice would be *m*=8 and any of the 30 primitive polynomials of that order [13].

8. KEP and ELGAMAL cipher protocols

It is necessary to define a protocol allowing for an interchange of information between Alice and Bob asynchronously to achieve the following objectives:

- deferred communications
- check the integrity of the exchanged information
- mutual authentication to avoid attacks from active adversaries (e.g. man-in-the-middle)
- block replay attacks
- availability of the exchanged information
- perfectly defined formats

The following protocol aims at fulfilling these requirements.

PROTOCOL: KEP AND CIPHER PUBLIC DATA EXCHANGES

INPUT: any kind of data to be exchanged between entities.
OUTPUT: encapsulated message (msg).
INITIAL CONFIGURATION:
msg: any kind of information to be exchanged between entities.
Universal-Keyed Message Authentication Code (UMAC): here proposed to assure strong symmetric authentication [3, 10].
ID: any elsewhere predefined and sender-receiver shared identification Tag.
K: sender-receiver shared key.
Tag: a smart label that can store any sort of information from identification numbers as a brief description for each entity. Here the tag is Tag = HMAC-SHA3-512 (HM || Nonce). See Fig 1 and more in [3]
HM : $NH_K(msg_1)$ || $NH_K(msg_2)$ || · · · || $NH_K(msg_r)$ || Len, see NH definition in [3].
Nonce: pseudorandom and unique number that changes with each generated tag.
Timestamp: formatted date and time.

MESSAGE AUTHENTICATION
1. **Acquire** K and msg
2. **Define** a fixed-length Nonce
3. **Generate** UMAC/SHA3-512/ID/Data
4. **Encapsulate** the concatenation of: [msg || UMAC/SHA3-512 (HM || Nonce) || Timestamp] into a file
5. **Send** the file and nonce to the receiver

MESSAGE VALIDATION
1. **Acquire** at any time the sent file
2. **Recover** msg and UMAC/SHA3-512 (HM || Nonce)
3. **Verify** integrity and sender's identity using K, Nonce, and msg
4. **Accept** or **Dismiss** msg according to the verification result

9. SECURITY AGAINST BRUTE FORCE ATTACKS

Lemma 2:

The complexity of attacking Algorithm 1 by brute force is given by $(ndim\ .\ q)^2.\ t$, where p is the shared prime , $q=\frac{p-1}{2}+1$ and ndim=rowsA.columnsA.

Proof:

To obtain the session key the attacker has to factorize matrices $U_k = A1_k\ .\ B1_k \pmod p$, k=1,…,t where the maximum value that can appear in every entry (i, j) is p-1, independently of ndim (see Algorithm 1). Thus, defining q=$\frac{p-1}{2}+1$; for each entry of each matrix, it is necessary to try q values, that is $(\mathrm{ndim}.q)^2$ possibilities. The attack is successful only if the factorization can be obtained for each k=1,…,t, something that can be easily avoided by choosing appropriate parameters ■

Example:
If p = 2147483647, ndim=100.90, t =10, then the complexity is ~ 2^{74}

10. Security against analytical attacks

The factorizations $U_k = A1_k \,.\, B1_k \pmod p$, k=1,…,t can be solved in $U_k \in R^{n\times n}$, $A1_k \in R^{n\times m}$, $B1_k \in R^{m\times n}$ (real realm), using the SVD (Singular Value Decomposition) if there are no restrictions regarding the nonnegativity of the factors, but if they are imposed as in our proposal, then Vavasis and references therein [25] proved that the problem is NP-hard in the continuous case. For the Boolean case, N. Gillis [9] wrote: *"...for exact factorizations, the rank-one problem is trivial. For higher ranks, Boolean factorization is equivalent to finding a rectangle covering of the matrix U. This is equivalent to the so-called biclique problem (given a bipartite graph defined by U, find the smallest number of complete bipartite subgraphs that cover the graph) which is NP-complete as proved in* [22]. *(sic)".* The Boolean NP-complete complexity was formally proven by Miettinen and Neumann [17].

11. Semantic security of algorithm 2

A formal description of this feature is given in APPENDIX A. Here we provide an informal view of this aspect, based on concepts mostly derived from Bellare's work [1]. In a nutshell, semantical security measures the resistance of any encryption algorithm to attacks using chosen plaintext or ciphertext selected by the attacker, who has access to the encryption and decryption modules working as oracles; without knowledge of the key selected for enciphering [1]. The semantic security term is strongly related to other definitions: the one-way functions and the non-malleability of ciphertexts.
Indistinguishability under chosen plaintext attack (represented as IND-CPA) is equivalent to the property of semantic security and is considered a basic requirement for most provably secure public-key cryptosystems [1]. One-way refers to bidirectional functions that have a probabilistic-polynomial time algorithm that converts domains into codomains, but no such algorithm is known that inverts the procedure. Non-malleability refers to the resistance to modify slightly the ciphertext to obtain meaningful recovered plaintext [1]. The next concept to define is the indistinguishability of different ciphertexts of two similar but different plaintexts, an attacker could not assign a ciphertext of one of them to any one of the plaintexts. This feature is generally presented as a game between a challenger (the algorithm defender) and an adversary (the algorithm attacker) [1]. The challenger generates a key pair PK, SK (public key and secret key respectively), based on any security parameter k (which can be the key size in bits), and publishes PK to the adversary. The challenger retains SK. Here we describe the adaptative version of the game. The adversary may perform any number of encryptions, decryptions, or any other operations. (The adversary is a probabilistic polynomial Turing Machine) [1]. Eventually, the adversary submits two distinct chosen plaintexts m_0 and m_1 to the challenger (of the same length). The challenger selects a bit $b \in \{0, 1\}$ uniformly at random, and sends the challenge ciphertext C = E (PK, m_b) back to the adversary. The adversary is free to perform any number of additional computations or decryptions (except C, this step is the adaptative

phase of the attack). Finally, its outputs in polynomial time a guess for the value of b [1]. The adversary wins the game if it guesses the bit b, and winning means the algorithm is not indistinguishable and secure, else the algorithm reaches the strongest available security level: IND-CCA2. (Indistinguishable chosen ciphertext adaptative attack). Formally, a cryptosystem is indistinguishable under an adaptative chosen ciphertext attack if no adversary can win the above game with probability p greater than $1/2+ \in_k$, where $\in_k \leq 1/\Pi^K$ (Π^K arbitrary polynomial function) and $\in_k$ is defined as a negligible function in the security parameter k [1]. For Algorithm 2 we prove the IND-CPA security level and explain how it could be easily adapted to reach the IND-CCA2 security level. The use of the UMAC function [3] in our Protocol fills this need in such a way that the practical implementation of Algorithms 1 and 2 culminates with the desired provable-security level.

12. NIST PQC security level of algorithm 2

NIST bases its classification on the range of security strengths offered by the existing NIST standards in symmetric cryptography, which NIST expects to offer significant resistance to quantum cryptanalysis. In particular, NIST defines a separate category for each of the following security requirements [21]. As previously described in the brute-force attack section, reasonable parameters exceed largely Level-1 PQC security.

13. A toy numerical example

Shared parameters: shared prime p = 5303, rowA=3, columnsA=2, t=2

ALICE

Iteration 1

Alice $A1_1 =$
$$\begin{matrix} 1123 & 341 \\ 14 & 238 \\ 1041 & 13 \end{matrix}$$

Alice $B1_1=$
$$\begin{matrix} 1525 & 1019 & 1561 \\ 1561 & 716 & 862 \end{matrix}$$

Alice $U_1 =$
$$\begin{matrix} 1707 & 4410 & 5290 \\ 446 & 4372 & 4284 \\ 1009 & 4184 & 2883 \end{matrix}$$

Iteration 2

Alice $A1_2 =$
$$\begin{matrix} 665 & 1338 \\ 622 & 38 \\ 505 & 1617 \end{matrix}$$

Alice $B1_2=$
$$\begin{matrix} 925 & 1412 & 598 \\ 364 & 463 & 409 \end{matrix}$$

Alice $U_2 =$
$$\begin{matrix} 4436 & 4695 & 978 \\ 549 & 4954 & 379 \\ 416 & 3406 & 3500 \end{matrix}$$

BOB

Iteration 1

Bob $A2_1 =$
$$\begin{matrix} 802 & 2435 \\ 1206 & 3408 \\ 707 & 3723 \end{matrix}$$

Bob $B2_1=$
$$\begin{matrix} 1174 & 2805 & 1242 \\ 3110 & 814 & 550 \end{matrix}$$

Bob $V_1 =$
$$\begin{matrix} 3083 & 5209 & 2014 \\ 3429 & 159 & 4847 \\ 4831 & 2322 & 3791 \end{matrix}$$

Iteration 2

Bob $A2_2 =$
$$\begin{matrix} 656 & 13 \\ 1900 & 107 \\ 611 & 1537 \end{matrix}$$

Bob $B2_2 =$
$$\begin{matrix} 2192 & 1270 & 845 \\ 820 & 1022 & 2194 \end{matrix}$$

Bob $V_2 =$
$$\begin{matrix} 893 & 3229 & 4815 \\ 4837 & 3429 & 117 \\ 1182 & 2858 & 1374 \end{matrix}$$

Alice computes A-KEY$_k$ = Det[A1$_k^T$ .V$_k$. B1$_k^T$] (mod p)   for k=1, 2

A-KEY$_1$ = 3207
A-KEY$_2$ = 2121
Alice's key =
0c3322f92446b51e3372d2a7bd2b81265bb96f32fa38562e4c02414e3c73d85ca4b358363b8792461d4033c1d76 23589c0f6c07ab01e33b6a7294019e125c779

Bob computes B-KEY$_k$ = Det[A2$_k^T$ .U$_k$. B2$_k^T$] (mod p)   for k=1, 2

B-KEY$_1$ = 3207
B-KEY$_2$ = 2121
Bob's key =
0c3322f92446b51e3372d2a7bd2b81265bb96f32fa38562e4c02414e3c73d85ca4b358363b8792461d4033c1d76 23589c0f6c07ab01e33b6a7294019e125c779

Example of the cipher algorithm

Bob ciphers a message to Alice

B-KEY = 32072121

PLAINTEXT msg (formatted as a 64-byte string with spaces appended on the right) = "***This is a secret communication.”***

CIPHERTEXT C =

```
    3083  5209  2014                  893 3229 4815
    3429   159  4847                 4837 3429   117
    4831  2322  3791                 1182  2858 1374
```

CIPHERTEXT D =
(88,91,75,138,4,47,198,62,82,82,161,194,222,89,228,82,123,218,0,95,151,77,56,71,47,99,53,39,83,29,246,124, 132,147,120,22,27,167,178,102,61,96,19,225,247,66,21,169,224,214,224,90,144,62,19,150,135,9,96,57,193,5, 231,89)

Alice recovers the message

A-KEY = 32072121

RECOVERED msg = “***This is a secret communication.”***

14. Numerical experiments

Programmed in RUST

OS: Windows 10 Pro (64 bits)

Processor: Intel(R) Core(TM) i7-3770K CPU @ 3.50GHz

RAM: 8,00 GB

The CPU times reported in the following Table 1 are the mean values of 10 runs for each combination of the variables, prime p = 2^{31}-1= 2147483647 (~31 bits)

rowsA	columnsA	Cycles	Complexity	CPU time in milliseconds
5	4	10	$\sim 2^{72}$	0.94
5	4	20	$\sim 2^{73}$	1.15
5	4	100	$\sim 2^{75}$	2.69
6	5	10	$\sim 2^{73}$	0.99
6	5	20	$\sim 2^{74}$	1.39
6	5	100	$\sim 2^{76}$	3.95
20	19	10	$\sim 2^{80}$	8.92
20	19	20	$\sim 2^{81}$	13.45
20	19	100	$\sim 2^{84}$	74.47
100	99	10	$\sim 2^{89}$	755.38
100	99	20	$\sim 2^{91}$	1503.54
100	99	100	$\sim 2^{93}$	7488.44

Table1. Algorithm 1 complexity and throughput time as a function

of the parameters.

Using 128 bits integers in RUST with prime p = 18446744073709551113 (~64 bits):

rowsA	columnsA	Cycles	Complexity	CPU time in milliseconds
5	4	10	$\sim 2^{138}$	1.29
6	5	10	$\sim 2^{139}$	0.98
20	19	10	$\sim 2^{146}$	26.84
100	99	10	$\sim 2^{156}$	3203.48

Table 2. Algorithm 1 complexity and throughput time as a function of the parameters.

15. Discussion

Algorithm 1 has been implemented in different computer languages and shows that extremely high complexity can be achieved in fractions of a second on a standard I7 processor. The fact that by modifying the input variables (number of rows, columns, primes, iterations) practically any security level can be easily obtained without resorting to multiple precision, leads to very fast implementations. Depending upon the computer architecture and software implementation, larger primes can be used for reaching higher complexity levels.
An easy enhancement – which will be reported in a forthcoming paper - is to use $\mathbb{F}_{2^m}$ polynomial fields, recurring to very fast field multiplications based on hard-coded discrete logarithm tables of any field generator base, on some steps. In such a way the lack of uniformity in the operations constitutes an additional barrier to possible algebraic attacks. This proposal could be very easily implemented without downgrading performance [4].
It is particularly important to use the UMAC function in the Protocol because it is similar to Merkle's trees for PQC digital signatures [2] and also plays the role of achieving maximal semantic security [1] and simultaneously strengthens its post-quantum character.
Note: as Black et al. [3] state "*the security of UMAC is rigorously proven, in the sense of giving exact and quantitatively strong results which demonstrate an inability to forge UMAC-authenticated messages assuming an inability to break the underlying cryptographic primitive. (sic)*"

16. Conclusions

The algorithms presented in this paper are such that very high complexity can be reached using small primes, normal precision, and small rectangular matrices, leading to very fast computer implementations.

Acknowledgments: to D. B. Szyld, L. Liberti, N. Gillis, J. Di Mauro, I. Córdoba, S. Barzola, G. Cucatti and Y. Alis for many interesting discussions, theoretical insights and computer implementations.

References

1. Bellare, M., Desai, A., Pointcheval, D., Rogaway, P.: Relations among notions of security for public-key encryption schemes. In Annual International Cryptology Conference, Springer, Berlin, Heidelberg, pp. 26-45 (1998)
2. Bernstein D., Lange T.: Post-Quantum Cryptography, Nature, 149,188-194 (2017)
3. Black, J., Halevi, S., Krawczyk, H., Krovetz, T., & Rogaway, P.: UMAC: Fast and secure message authentication, Annual International Cryptology Conference, pp. 216-233, Springer, Berlin, Heidelberg (1999)
4. Daemen, J., Rijmen, V.: AES Proposal: Rijndael, AES algorithm submission, September 3 (1999)
5. Di Mauro, J., Salazar, E. & Scolnik, H.D.: Design and implementation of a novel cryptographically secure pseudorandom number generator. J Cryptogr Eng, 12, 255–265 https://doi.org/10.1007/s13389-022-00297-8 (2022)
6. Diffie, W., Hellman, M.: New directions in cryptography", IEEE Transactions on Information Theory, 22, 6, 644-654 (1976)
7. Ellis, J. H.: The possibility of non-secret digital encryption, CESG Research Report (1970)
8. Fujisaki, E., Okamoto, T.: Secure Integration of Asymmetric and Symmetric Encryption Schemes, in M. Wiener (Ed.): CRYPTO'99, LNCS 1666, Springer-Verlag (1999)
9. Gillis N., Private communication (2022)
10. Internet Engineering Task Force (IETF), UMAC RFC 4418, https://datatracker.ietf.org/doc/rfc4418/ (2006). Accessed 15 September 2022
11. Kanwal, S., Ali, R.: A cryptosystem with noncommutative platform groups. Neural Computing and Applications, 29: 11, 1273-1278 (2018).
12. Lee, G.T.: Abstract Algebra, Springer Undergraduate Mathematics Series, https://doi.org/10.1007/978-3-319-77649-1_3 (2018)
13. Lidl, R., Niederreiter, H.: Introduction to Finite Fields and their Applications, Cambridge University Press, Cambridge (1997)
14. Liu, J., Jia, J., Zhang, H., Yu, R., Yu, Y., Wu, W.: Cryptanalysis of a cryptosystem with non-commutative platform groups. China Communications, 15(2), 67-73 (2018)
15. Maurer, U., Renner, R., Holenstein, C.: Indifferentiability, Impossibility Results on Reductions, and Applications to the Random Oracle Methodology, M. Naor (Ed.): TCC 2004, LNCS 2951, pp. 21–39, Springer-Verlag, (2004)
16. Menezes, A., Van Oorschot, P., Vanstone, S.: Handbook of applied cryptography. The CRC Press series on discrete mathematics and its applications, CRC-Press (1997)
17. Miettinen, P., Neumann, S.: Recent Developments in Boolean Matrix Factorization, Ninth International Joint Conference on Artificial Intelligence (IJCAI-20), a slightly extended version of the survey is available at preprint https://doi.org/10.48550/arXiv.2012.03127 (2020)
18. Mullan, C.: Some Results in Group-Based Cryptography, Thesis submitted to the University of London for the Degree of Doctor of Philosophy (2020)
19. Myasnikov, A., Shpilrain, V., Ushakov A.: Non-commutative Cryptography and Complexity of Group-theoretic Problems, Mathematical Surveys and Monographs, AMS Volume 177 (2011)
20. NIST Computer Security Resource Center: Competition for standardization of post-quantum protocols (PQC), https://csrc.nist.gov/projects/post-quantum-cryptography (2017). Accessed 15 September 2022
21. NIST Computer Security Resource Center: Post-Quantum Security, https://csrc.nist.gov/projects/post-quantum-cryptography/post-quantum-cryptography-standardization/evaluation-criteria/security-(evaluation-criteria) (2022). Accessed 15 September 2022
22. Peeters R.: The maximum edge biclique problem is NP-complete, Discrete Applied Mathematics, Volume 131, Issue 3, pp 651-654 (2003)
23. Rotman, J. J.: Advanced Modern Algebra, vol. 114, American Mathematical Soc. (2010)
24. Stickel, E.: A new method for exchanging secret keys. Proceedings of the Third International Conference on Information Technology and Applications (ICITA05), Contemporary Mathematics, IEEE Computer Society, 2, 426–430 (2005)
25. Vavasis, S.: On the complexity of nonnegative matrix factorization, SIAM J. Optim., 20, pp. 1364–1377 (2010),
26. Virdia F.: private communication (2022)

APPENDIX A: SEMANTIC SECURITY OF ALGORITHM 2

Definitions:

Security levels are usually defined by pairing each goal (OW: one-way, IND: indistinguishability, NM: non-malleability) with an attack model like IND-CPA or IND-CCA2 [1].

Non-commutative rectangular matrices structure (NCRMS): the regular multiplication of non-square matrices is only defined if the number of columns of the first factor equals the number of rows of the second one and non-square matrices are in general non-commutative, concerning regular multiplication in Z_p, except with Hadamard multiplication of equal dimensioned matrices [12, 22]. This is the irregular structure used in algorithm 2.

FSP problem over NCRMS: Assuming an attack against U_k (alternatively V_k), any vector term individually token, we define the Factorization-Search Problem (FSP) [18] over non-commutative structures as the one-way trapdoor function (OWTF), assumed to be computationally intractable, as follows:

FSP: *given a matrix $U^*_{k,}$, find the rectangular matrices $A1_k$, and $B1_k$ such that $A1_k$. $B1_k$ (mod p) = U^*_k.*

Note: FSP problem over NCRMS is NP-complete as exposed before at Point 10 security against analytical attacks.

Theorem 1. For a plaintext message uniformly distributed in the plaintext message space, the algorithm 2 cryptosystem is all-or-nothing secure against CPA under the FSP assumption if H( ) hash function is a random oracle [15].

Proof. On the one hand, if the FSP problem is tractable over NCRMS, for any given ciphertext pair (C, D), it is easy to compute a-key[k], repeating the procedure t-times, and then extract the plaintext

$$\text{msg} = \text{H(A-CONCAT)} \oplus \text{D}.$$

On the other hand, suppose there exists an efficient adversary A with access to the random oracle H against the above cryptosystem; that is, given public or openly transmitted data and ciphertext then A outputs msg with a non-negligible advantage s.

If the adversary's advantage s is non-negligible, he makes an H-query on a trial A-CONCAT and recovers the msg, Otherwise, if H is modeled as a cryptographic secure hash algorithm [15], A's advantage should be negligible no matter what he can compute before making such a query.

Thus, we can solve the FSP problem over NCRMS with the non-negligible probability s. This contradicts the holding of the FSP assumption. ■

The above theorem says that algorithm 2 reaches the weak security, i.e. the IND-CPA, but using the technique due to Fujisaki-Okamoto [8], the protocol could be easily converted to IND-CCA2 security, using random-salt concatenation of the msg.