PQC: R-Propping of Burmester-Desmedt Conference Key Distribution System

Pedro Hecht

Information Security Master, School of Economic Sciences,
School of Exact and Natural Sciences and Engineering School (ENAP-FCE),
University of Buenos Aires, Av. Cordoba 2122 2nd Floor,
CABA C1120AAP, República Argentina
phecht@dc.uba.ar

Abstract. Post-quantum cryptography (PQC) is a trend that has a deserved NIST status, and which aims to be resistant to quantum computer attacks like Shor and Grover algorithms. NIST is currently leading the third-round search of a viable set of standards, all based on traditional approaches as code-based, lattice-based, multi quadratic-based, or hash-based cryptographic protocols [1]. We choose to follow an alternative way of replacing all numeric field arithmetic with GF(2^8) field operations [2]. By doing so, it is easy to implement R-propped asymmetric systems as the present paper shows [3,4]. Here R stands for Rijndael as we work over the AES field. This approach yields secure post-quantum protocols since the resulting multiplicative monoid is immune against quantum algorithms and resist classical linearization attacks like Tsaban's Algebraic Span [5] or Roman'kov linearization attacks [6]. The Burmester-Desmedt (B-D) conference key distribution protocol [7] has been proved to be secure against passive adversaries if the computational Diffie-Hellman problem remains hard. The authors refer that the proposed scheme could also be secure against active adversaries under the same assumptions as before if an authentication step is included to foil attacks like MITM (man in the middle). Also, this protocol proved to be semantical secure against adaptative IND-CPA2 [8, 9] if the discrete log problem is intractable. We discuss the features of our present work and a practical way to include an authentication step. Classical and quantum security levels are also discussed. Finally, we present a numerical example of the proposed R-Propped protocol.

Keywords: Post-quantum cryptography, conference key distribution, finite fields, combinatorial group theory, R-propping, public-key cryptography, non-commutative cryptography, AES.

1 Introduction

1.1 PKC Proposals Based on Combinatorial Group Theory

The theoretical foundations for the current generation of cryptosystems lie in the intractability of problems close to number theory [10] and therefore prone to quantum attacks. This was the main reason to develop PQC. It is noteworthy that besides a couple of described solutions [1], there remain overlooked solutions belonging to non-commutative (NCC) and non-associative (NAC) algebraic cryptography [10]. The general structure of these solutions relies on protocols defining one-way trapdoor functions (OWTF) extracted from the combinatorial group theory [11].

1.2 The motivation of the present work

In this paper, we apply our algebraic patch [2]to the well-known Burmester-Desmedt (B-D) conference key distribution [7]. In essence, it is a generalization of Diffie-Hellman two parties protocol [12] to an undefined number of entities while maintaining the number of interchanges constant. That protocol has the virtue of presenting a proved semantic secure systems attaining IND-CPA2 level as long computational Diffie-Hellman and discrete log problems hold. The main target is to make that protocol quantum resistant.

Essentially R-propping consists of replacing all numerical field operations (arithmetic sum and multiplication), a typical scalar proposal, by algebraic operations using the AES field, a vectorial proposal [2]. This scales up operations complexity foiling classical linearization attacks, like AES [13] does and at the same time quantum ones. This is a solid way to achieve the best of two worlds, both pointing to cryptographic security. As side benefits, we get rid of big number libraries and step away from the critical dependency of pseudo-random generators.

The R-propping solution is described as an Algebraic Extension Ring (AER) [2]. For background knowledge about algebraic solutions, we refer to the Myasnikov NCC treatise [11] which contributes to exhaustive knowledge of the cryptographic application of the combinatorial field theory.

2 Preliminaries

Definition 1 (Security levels). Currently, there are several types of attack models for public-key encryption, namely the chosen-plaintext attack (CPA), non-adaptive chosen-ciphertext attacks (CCA1), and adaptive chosen-ciphertext attacks (CPA2, CCA2). Security levels are usually defined by pairing each goal (2: adaptative version, OW: one-way, IND: indistinguishability, NM: non-malleability) with an attack model (CPA, CCA1 or CPA2, CCA2); i.e., OW-CPA, OW-CCA1, OW-CCA2; IND-CPA, IND-CPA2, IND-CCA1 and IND-CCA2 [8, 9].

Definition 2 (Algebraic Extension Ring - AER). The Algebraic Extension Ring (AER) framework includes the following structures:

$\mathbb{F}_{256}$: a.k.a. *GF*[2^8], the AES field [6]

Primitive polynomial: $1+x+x^3+x^4+x^8$ with <*1+x*> as the multiplicative subgroup ($\mathbb{F}^*_{255}$) generator:

M[$\mathbb{F}_{256}$ d] d-dimensional square matrix of field elements. (bytes). Therefore, a d-dimensional square matrix is equivalent to a rank-3 Boolean tensor.

The AER platform has two substructures:

$(M[\mathbb{F}_{256}, d], \oplus, 0)$ Abelian group using field sum as operation and null matrix (tensor) as the identity element.

$(M[\mathbb{F}^*_{255}, d], \odot, I)$ Non-commutative monoid using field product as operation and identity matrix (tensor) as the identity element.

From here on, when referring to field elements (bytes) we call them simply as elements, and when we refer to any d-dimensional matrix of the AER we will use the term d-dim tensor.

Detailed information on AER could be read at [2].

Definition 3 (One-Way Trapdoor Functions – OWTF): these are the core of the canonical protocols for asymmetric cryptography based on the combinatorial group theory. They are based on hard problems, traditionally using commutative numeric fields, but the same problem definitions could be applied to non-commutative monoids (as in AER) :

- **Computational Diffie-Hellman Problem** (CDHP): Given (z1, z2) ϵ Z^2 and x ϵ AER, compute $x^{z1z2}= x^{z2z1}$ for given x, x^{z1}, and x^{z2}.

– **Discrete Logarithm Problem** (DLP): Given z ϵ Z and x ϵ AER, compute z for given x and x^z.

For general non-commutative structure like the multiplicative monoid of AER, the above problems are difficult enough to be cryptographic assumptions, meaning that there does not exist a probabilistic polynomial-time algorithm that can solve all instances of them with non-negligible accuracy concerning the problem scale, i.e., the number of input bits of the problem).

3 Burmester-Desmedt (B-D) distributed the conference key.

Burmester and Desmedt protocol is carried out by composing n-participants in a ring structure. An example of four entities is performed through the stages of Table 1.

ALICE	BOB	CHARLIE	DAVID
Public prime p, generator <g>			
Private a, g^a → to D, to B	Private b, g^b → to A, to C	Private c, g^c → to B, to D	Private d, g^d → to C, to A
Public Xa = $(g^b/g^d)^a$ Private Za=g^{ad}	Public Xb = $(g^c/g^a)^b$ Private Zb=g^{ab}	Public Xc = $(g^d/g^b)^c$ Private Zc=g^{bc}	Public Xd = $(g^a/g^c)^d$ Private Zd=g^{cd}
Private Ka= $Za^4Xa^3Xb^2Xc$	Private Kb= $Zb^4Xb^3Xc^2Xd$	Private Kc= $Zc^4Xc^3Xd^2Xa$	Private Kd= $Zd^4Xd^3Xa^2Xb$
Ka=Kb=Kc=Kd			

Table 1. A schematic view of the original Burmester-Desmedt conference key distribution protocol for a small ring of n=4 entities. This protocol involves a double pass exchange. The session key is a cyclic but not symmetric function of degree two.

4 R-Propped B-D distributed conference key.

The differences between the original and the R-Propped version are:

1. Instead of a cyclic (commutative) multiplicative group structure Z^*_p in a numeric field, we work over the non-commutative multiplicative monoid of the algebraic extension ring (AER) defined at point 2. Preliminaries.
2. The elements of AER are d-dimensional square matrices (referred to as tensors) of F_{256} field elements. Sums and products of tensors are field operations.
3. The generator <G> is a predefined non-singular tensor G. The period |<G>| of the cyclic subgroup is empirically obtained through computational simulation.
4. Inverses of tensors are obtained through exponentiation using the period |<G>| minus one. The |<G>| power of each generator is the identity matrix.

5 The cryptographic security of R-propped B-D protocol

The security of the protocol relies on the intractability of CDHP and DLP problems. Using R-Propping we design private keys (exponents) of certain public tensors for which this approach is unfeasible.

The proposed public generators are:

dim 3, period 256^3 --> 2^24

$$G3 = \begin{pmatrix} 158 & 215 & 6 \\ 216 & 221 & 53 \\ 45 & 119 & 206 \end{pmatrix}$$

dim 4, period 256^4 --> 2^32

$$G4 = \begin{pmatrix} 210 & 72 & 68 & 31 \\ 156 & 225 & 86 & 224 \\ 75 & 171 & 53 & 252 \\ 38 & 22 & 171 & 109 \end{pmatrix}$$

dim 7, period 256^12 --> 2^96

$$G7 = \begin{pmatrix} 147 & 65 & 106 & 219 & 36 & 20 & 37 \\ 125 & 14 & 216 & 138 & 90 & 186 & 10 \\ 67 & 90 & 56 & 25 & 234 & 130 & 86 \\ 156 & 242 & 122 & 74 & 146 & 218 & 128 \\ 19 & 55 & 159 & 189 & 5 & 142 & 114 \\ 236 & 247 & 81 & 75 & 124 & 61 & 121 \\ 119 & 15 & 112 & 21 & 195 & 25 & 118 \end{pmatrix}$$

dim 10, period 256^14 --> 2^112

$$G10 = \begin{pmatrix} 222 & 179 & 28 & 115 & 147 & 20 & 69 & 102 & 39 & 46 \\ 233 & 103 & 227 & 60 & 170 & 63 & 13 & 0 & 203 & 20 \\ 70 & 52 & 2 & 77 & 155 & 51 & 203 & 221 & 185 & 27 \\ 234 & 69 & 0 & 3 & 113 & 112 & 137 & 237 & 143 & 140 \\ 92 & 243 & 15 & 70 & 59 & 75 & 141 & 157 & 213 & 251 \\ 75 & 208 & 88 & 243 & 83 & 17 & 130 & 10 & 129 & 4 \\ 241 & 97 & 241 & 224 & 192 & 213 & 105 & 53 & 232 & 226 \\ 41 & 15 & 123 & 22 & 144 & 73 & 111 & 228 & 191 & 15 \\ 83 & 131 & 155 & 183 & 158 & 84 & 183 & 144 & 189 & 78 \\ 126 & 35 & 224 & 17 & 157 & 124 & 32 & 140 & 118 & 226 \end{pmatrix}$$

dim 12, period 256^20 --> 2^160

G12 =

$$\begin{pmatrix} 255 & 21 & 43 & 199 & 233 & 44 & 168 & 110 & 205 & 105 & 190 & 140 \\ 254 & 241 & 192 & 46 & 189 & 239 & 112 & 129 & 236 & 114 & 30 & 162 \\ 78 & 182 & 117 & 99 & 1 & 213 & 173 & 144 & 178 & 105 & 22 & 104 \\ 235 & 237 & 38 & 152 & 100 & 43 & 160 & 194 & 10 & 230 & 21 & 237 \\ 29 & 127 & 72 & 1 & 236 & 4 & 152 & 37 & 13 & 125 & 205 & 108 \\ 55 & 159 & 168 & 196 & 238 & 6 & 139 & 43 & 155 & 146 & 100 & 112 \\ 133 & 25 & 117 & 59 & 130 & 198 & 212 & 87 & 109 & 42 & 105 & 147 \\ 147 & 254 & 177 & 199 & 205 & 140 & 60 & 115 & 72 & 225 & 7 & 45 \\ 198 & 136 & 42 & 71 & 13 & 95 & 115 & 146 & 195 & 245 & 68 & 31 \\ 239 & 56 & 211 & 16 & 19 & 67 & 207 & 229 & 203 & 155 & 94 & 105 \\ 41 & 182 & 182 & 57 & 223 & 173 & 161 & 246 & 32 & 71 & 233 & 120 \\ 17 & 43 & 171 & 195 & 86 & 58 & 255 & 237 & 158 & 65 & 84 & 9 \end{pmatrix}$$

Table 2. Predefined tensors <G> and corresponding multiplicative orders to be used for the B-D protocol.

Classical and quantum security levels are as follows:

Tensor dimension	<G> proposed generator	Period |<G>|	Classical Security (bits)	[Grover] Quantum Security (bits)
3	G3	2^{24}=16777216	24	12
4	G4	2^{32} = 4294967296	32	16
7	G7	2^{96} = 7.92 x 10^{28}	96	48
10	G10	2^{112} = 5.19 x 10^{33}	112	56
12	G12	2^{160}= 1.46 x 10^{48}	160	80

Table 3. Expected security of increasing size of private keys subject to classical and quantum attacks. Depending on the particular situation, it should be chosen security parameters like G7 or above.

The IND-CPA2 semantic security is assured as members of the <G> set are indistinguishable from random tensors of the same size. Statistic evidence of tensor structures is provided at [4]. As this protocol is susceptible to a MITM attack, it is convenient to include an authentication step including public key certificates or HMAC of session keys with public ID values.

6 Step-By-Step Example

To follow procedures, we show a dim=3 toy program written for Mathematica 12 interpreted language. Detailed code with the newly defined functions is available upon request to the author. Running as-is on an Intel®Core™i5-5200U CPU 2.20 GHz the registered mean session time was 4.40 s.

(v.3) Errata> at the below-printed program, variable "period" was undefined and should be set as period = 2^24 – 1, instead of the mentioned 2^24. The same error appears at the output, but it has not affected further operations since the correct value was retained as an environmental memory value.

```
Print["R-PROPPED BURMESTER-DESMEDT CONFERENCE KEY DISTRIBUTION"];
Print["Small dimension step-by-step example"];
Print["......................................................."];
Print["PUBLIC PARAMETERS......................................"];
Print["n=4 entities ring: -->ALICE-->BOB-->CHARLIE-->DAVID-->"];
dim = 3; Print["tensor dim=", dim];
zlimit = 2^24 - 1
Print["period = ", 2^24];
Print["maximum exponent=", zlimit];
Label[begin];
     158 215   6
G = (216 221  53 ); Print["tensor G3=", MatrixForm[G]];
      45 119 206
If[Tdet3[G] == 0, Goto[begin],] (* non singular *)
iG = TFastPower[G, period - 1];
If[TProd[iG, G] == IdentityMatrix[3], , Goto[begin]]
(* true inverse *)

Print["PRIVATE EXPONENTS....................................."];
a = RandomInteger[{1, zlimit}]; Print["ALICE   a=", a];
b = RandomInteger[{1, zlimit}]; Print["BOB     b=", b];
c = RandomInteger[{1, zlimit}]; Print["CHARLIE c=", c];
d = RandomInteger[{1, zlimit}]; Print["DAVID   d=", d];

Print["FIRST TOKEN..........................................."];
Ga = TFastPower[G, a]; Print["ALICE   Ga=", MatrixForm[Ga]];
iGa = TFastPower[Ga, period - 1]; (*inverse of Ga*)
Gb = TFastPower[G, b]; Print["BOB     Gb=", MatrixForm[Gb]];
iGb = TFastPower[Gb, period - 1]; (*inverse of Gb*)
Gc = TFastPower[G, c]; Print["CHARLIE Gc=", MatrixForm[Gc]];
iGc = TFastPower[Gc, period - 1]; (*inverse of Gc*)
Gd = TFastPower[G, d]; Print["DAVID   Gd=", MatrixForm[Gd]];
iGd = TFastPower[Gd, period - 1]; (*inverse of Gd*)

Print["SECOND TOKEN.........................................."];
Xa = TFastPower[TProd[Gb, iGd], a];
Print["ALICE   Xa=", MatrixForm[Xa]];
Za = TFastPower[Gd, a];
Xb = TFastPower[TProd[Gc, iGa], b];
Print["BOB     Xb=", MatrixForm[Xb]];
Zb = TFastPower[Ga, b];
Xc = TFastPower[TProd[Gd, iGb], c];
Print["CHARLIE Xc=", MatrixForm[Xc]];
Zc = TFastPower[Gb, c];
Xd = TFastPower[TProd[Ga, iGc], d];
Print["DAVID   Xd=", MatrixForm[Xd]];
Zd = TFastPower[Gc, d];
```

```
Print["CONFERENCE KEY......................................."]
(* Ka = Za^4 . Xa^3 . Xb^2 . Xc  *)
Za4 = TFastPower[Za, 4];
Xa3 = TFastPower[Xa, 3];
Xb2 = TFastPower[Xb, 2];
P1 = TProd[Xb2, Xc];
P2 = TProd[P1, Xa3];
Ka = TProd[P2, Za4];
Print["ALICE   Ka=", MatrixForm[Ka]];
(* Kb = Zb^4 . Xb^3 . Xc^2 . Xd  *)
Zb4 = TFastPower[Zb, 4];
Xb3 = TFastPower[Xb, 3];
Xc2 = TFastPower[Xc, 2];
P1 = TProd[Xc2, Xd];
P2 = TProd[P1, Xb3];
Kb = TProd[P2, Zb4];
Print["BOB     Kb=", MatrixForm[Kb]];
(* Kc = Zc^4 . Xc^3 . Xd^2 . Xa  *)
Zc4 = TFastPower[Zc, 4];
Xc3 = TFastPower[Xc, 3];
Xd2 = TFastPower[Xd, 2];
P1 = TProd[Xd2, Xa];
P2 = TProd[P1, Xc3];
Kc = TProd[P2, Zc4];
Print["CHARLIE Kc=", MatrixForm[Kc]];
(* Kd = Zd^4 . Xd^3 . Xa^2 . Xb  *)
Zd4 = TFastPower[Zd, 4];
Xd3 = TFastPower[Xd, 3];
Xa2 = TFastPower[Xa, 2];
P1 = TProd[Xa2, Xb];
P2 = TProd[P1, Xd3];
Kd = TProd[P2, Zd4];
Print["DAVID  Kd=", MatrixForm[Kd]];
Print["......................................................"];
If[Ka == Kb == Kc == Kd,
  Print["Validated conference key"], GoTo [begin]];
Print["......................................................"];
```

And the corresponding output is:

```
R-PROPPED BURMESTER-DESMEDT CONFERENCE KEY DISTRIBUTION
Small dimension step-by-step example
.......................................................
PUBLIC PARAMETERS......................................
n-4 entities ring: -->ALICE-->BOB-->CHARLIE-->DAVID-->
tensor dim-3
16 777 215
period - 16 777 216
maximum exponent-16 777 215
           ⎛158  215   6  ⎞
tensor G3-  216  221  53
           ⎝ 45  119  206 ⎠
PRIVATE EXPONENTS......................................
ALICE   a-13 268 292
BOB     b-3 256 521
CHARLIE c-14 378 566
DAVID   d-16 302 982
FIRST TOKEN............................................
            ⎛228  227  104 ⎞
ALICE   Ga-  124  102  50
            ⎝ 96   46  186 ⎠

            ⎛123  229  218 ⎞
BOB     Gb-  190  64   176
            ⎝162  192  169 ⎠

            ⎛ 11   41   48 ⎞
CHARLIE Gc-  190  145  79
            ⎝ 92   50  110 ⎠

            ⎛251   95   70 ⎞
DAVID   Gd-  162  187  97
            ⎝ 37  155  182 ⎠
SECOND TOKEN...........................................
            ⎛ 72   80    4 ⎞
ALICE   Xa-   36  145   7
            ⎝165   87  154 ⎠

            ⎛240   98  160 ⎞
BOB     Xb-   94  135  74
            ⎝147   58    6 ⎠

            ⎛219   58  183 ⎞
CHARLIE Xc-  124  175  22
            ⎝153   60  244 ⎠

            ⎛177  139   75 ⎞
DAVID   Xd-  153  65   174
            ⎝108  227  246 ⎠
CONFERENCE KEY.........................................
            ⎛35  48  147 ⎞
ALICE   Ka-   7  243  14
            ⎝63  165 111 ⎠

            ⎛35  48  147 ⎞
BOB     Kb-   7  243  14
            ⎝63  165 111 ⎠

            ⎛35  48  147 ⎞
CHARLIE Kc-   7  243  14
            ⎝63  165 111 ⎠

           ⎛35  48  147 ⎞
DAVID  Kd-   7  243  14
           ⎝63  165 111 ⎠
.......................................................
Validated conference key
.......................................................
```

7 Conclusions

We present a PQC class solution to the distributed conference key necessity. Practical parameters are presented, and they solve the central question with different security levels.

Other works of the author covering this field can be found at [14].

References

1. D. J. Bernstein, T. Lange, "Post-Quantum Cryptography", Nature, 549:188-194, 2017
2. P. Hecht, Algebraic Extension Ring Framework for Non-Commutative Asymmetric Cryptography, https://arxiv.org/ftp/arxiv/papers/2002/2002.08343.pdf 1.2, 2020
3. P. Hecht, R-Propping of HK17: Upgrade for a Detached Proposal of NIST PQC First Round Survey, https://eprint.iacr.org/2020/1217, 2020
4. P. Hecht, PQC: R-Propping of Public-Key Cryptosystems Using Polynomials over Non-commutative Algebraic Extension Rings, https://eprint.iacr.org/2020/1102, 2020
5. A. Ben Zvi, A. Kalka, B. Tsaban, Cryptanalysis via algebraic spans, CRYPTO 2018, Lecture Notes in Computer Science 10991 255-274. https://doi.org/10.1007/978-3-319-96884-1_9, 2018
6. V. Roman'kov, Cryptanalysis of a combinatorial public key crypto-system, DeGruyter, Groups Complex. Cryptology. 2017.
7. M. Burmester and Y. Desmedt, A Secure and Efficient Conference Key Distribution System. In A. De Santis, editor, Advances in Cryptology, EUROCRYPT'94, volume 950 of Lecture Notes in Computer Science, pages 275–286. Springer, 1995
8. E. Kiltz, J. Malone-Lee, A General Construction of IND-CCA2 Secure Public Key Encryption, ruhr-uni-bochum.de/Eike.Kiltz/papers/general_cca2.ps
9. S. Goldwasser, S. Micali, "Probabilistic Encryption", Journal of Computer and System Sciences, 28: 270-299, 1984.
10. A. Menezes, P. van Oorschot and S.Vanstone, "Handbook of Applied Cryptography", CRC Press, 1997
11. A. Myasnikov, V. Shpilrain, A. Ushakov, Non-commutative Cryptography and Complexity of Group-theoretic Problems, Mathematical Surveys and Monographs, AMS Volume 177, 2011
12. W. Diffie and M.E. Hellman, New directions in cryptography, IEEE Transactions on Information Theory 22 (1976), 644-654. 1.1, 4.3
13. FIPS PUB 197: the official AES standard, https://web.archive.org/web/20150407153905/http://csrc.nist.gov/publications/fips/fips197/fips-197.pdf
14. https://arxiv.org/a/hecht_p_1.html