PQC: R-Propping of a New Group-Based Digital Signature

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 [1]. We choose to follow a non-standard way to achieve PQC: taking any standard asymmetric protocol and replacing numeric field arithmetic with GF(2^8) field operations [2]. By doing so, it is easy to implement R-propped asymmetric systems as present and former papers show [3,4,5]. Here R stands for Rijndael as we work over the AES field. This approach yields secure post-quantum protocols since the resulting multiplicative monoid resists known quantum algorithm and classical linearization attacks like Tsaban's Algebraic Span [6] or Roman'kov linearization attacks [7]. Here we develop an original group-based digital signature protocol and R-propped it. The protocol security relies on the intractability of a generalized discrete log problem, combined with the power sets of algebraic ring extension tensors [2]. The semantic security and classical and quantum security levels are discussed. Finally, we present a numerical example of the proposed protocol.

Keywords: Post-quantum cryptography, finite fields, combinatorial group theory, R-propping, public-key cryptography, non-commutative cryptography, digital signature, IND-CCA2,

1. Introduction

1.1. PQC Proposals Based on Combinatorial Group Theory

Besides currently evaluated PQC solutions like code-based, hash-based, multi quadratic, or lattice-based cryptography, there remain overlooked solutions belonging to non-commutative (NCC) and non-associative (NAC) algebraic cryptography. The general structure of these solutions relies on one-way trapdoor functions (OWTF) extracted from the combinatorial group theory [8].

1.2. Motivation of the present work

In this paper, we develop an algebraic digital signature protocol The main target is to achieve quantum-attacks resistance.

R-propping consists of replacing numerical field operations with algebraic operations using the AES field [2]. As a benefit, no big number libraries are needed, and eradicating the critical dependency on pseudo-random generators that affects protocols that security relies on big prime numbers.

The R-propping solution is described below as an Algebraic Extension Ring (AER). For background knowledge about algebraic solutions, we refer to the Myasnikov et al NCC treatise [8].

2. Background

2.1. Algebraic Extension Ring (AER). The algebraic extension ring framework [2] includes the following structures:

$\mathbb{F}_{256}$: a.k.a. *GF*[2^8], the AES (advanced encryption standard) field [9]

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], $\oplus$, O) 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 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].

2.2. Generalized discrete logarithm problem (GDLP) in AER framework.

Given $t_2=(t_1)^x$, where t_1 is an unknown tensor and x an unknown integer, compute exponent x for a given t_2 tensor.

3. R-Propped group-based digital signature protocol

It is proposed an indirect signature procedure, so a suitable public hashing of a binary message msg of arbitrary length n should be defined. We choose a numeric output h(msg) =h($\{0, 1\}^n$) ϵ [1, period]. A period is defined at 3.1. This function should be publicly available together with the tensor product and power functions. The protocol uses the AES framework [9]. The implementation takes the following steps:

3.1. Define the desired security level from Table 2. , selecting the corresponding base generator g_0 and period and using the numeric definition in Table 1. This g_0 and period are both public data.

3.2. Any signer defines his msg and compute h(msg).

3.3. The signer generates a random secret exponent r in the range [2, period-2] and computes the r-power of g_0. This will be the actual private generator g. Then he computes a random session private key (a) in the range [2, period-2] and the corresponding public key and the first component of the digital signature $s_0=(g)^a$

3.4. The signer computes the inverse tensor g^{-1} raising g to power period -1 and control that the product $g.g^{-1}$ = identity tensor. If not, returns to 3.3.

3.5. The signer defines a secret session key k in the range [2, period-2] and computes the exponent kh = k . h, where h=h(msg).

3.6. The signer compute the signatures $s_1= s_0\ (g^{-1})^{k.h}$ and $s_2=(g)^k$.

3.7. The signer publishes the digital signature (s_0, s_1, s_2) together with the message msg.

3.8. Any verifier should:

3.8.1. Using the msg, recalculate h'=h(msg)

3.8.2. compute the power $s_3=(s_2)^{h'}$

3.8.3. Verify if the product s1.s3 = s0. If true, then the computed h'= h(msg) matches the original h=h(msg), so the signature is valid, else it would be rejected.

3.9. If verified, the origin of the signature, the integrity of message, and non-repudiation are assured.

4. The cryptographic security of the R-Propped B-D protocol

Using R-Propping we design private keys (exponents) of certain public tensors for which this approach is unfeasible.

The proposed tensor generators are:

dim 3, period 256^3 – 1 -- > 2^24 – 1

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

dim 4, period 256^4 – 1 -- > 2^32 – 1

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

dim 7, period 256^12 – 1 -- > 2^96 – 1

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

dim 10, period 256^14 – 1 -- > 2^112 – 1

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

dim 12, period 256^20 – 1-- > 2^160 – 1

$$G12 = \begin{pmatrix} 255 & 21 & 43 & 199 & 233 & 44 & 168 & 110 & 205 & 105 & 190 & 140 \\ 254 & 241 & 192 & 46 & 189 & 239 & 112 & 129 & 236 & 114 & 30 & 162 \\ 78 & 182 & 117 & 99 & 1 & 213 & 173 & 144 & 178 & 105 & 22 & 104 \\ 235 & 237 & 38 & 152 & 100 & 43 & 160 & 194 & 10 & 230 & 21 & 237 \\ 29 & 127 & 72 & 1 & 236 & 4 & 152 & 37 & 13 & 125 & 205 & 108 \\ 55 & 159 & 168 & 196 & 238 & 6 & 139 & 43 & 155 & 146 & 100 & 112 \\ 133 & 25 & 117 & 59 & 130 & 198 & 212 & 87 & 109 & 42 & 105 & 147 \\ 147 & 254 & 177 & 199 & 205 & 140 & 60 & 115 & 72 & 225 & 7 & 45 \\ 198 & 136 & 42 & 71 & 13 & 95 & 115 & 146 & 195 & 245 & 68 & 31 \\ 239 & 56 & 211 & 16 & 19 & 67 & 207 & 229 & 203 & 155 & 94 & 105 \\ 41 & 182 & 182 & 57 & 223 & 173 & 161 & 246 & 32 & 71 & 233 & 120 \\ 17 & 43 & 171 & 195 & 86 & 58 & 255 & 237 & 158 & 65 & 84 & 9 \end{pmatrix}$$

Table 1. Predefined base tensors <G_0> and corresponding multiplicative orders to be used for the R-Propped protocol: any base tensor raised to the corresponding period yields the Identity tensor. This table redefines Table 2. published in [5].

Classical and quantum security levels are as follows:

Tensor dimension	<G_0> base generator	cyclic period |<G>|	Classical Security (bits)	[Grover] Quantum Security (bits)
3	G3	2^{24} - 1=16777215	24	12
4	G4	2^{32} - 1= 4294967295	32	16
7	G7	2^{96} - 1= 7.92 x 10^{28}	96	48
10	G10	2^{112} - 1= 5.19 x 10^{33}	112	56
12	G12	2^{160} - 1= 1.46 x 10^{48}	160	80

Table 2. Expected security of increasing size of private keys subject to classical and quantum attacks. Depending on the situation, it should be chosen base generators like G7 or above from Table 1. In any case, any random power of the base generator should be used as the actual generator of the protocol. This table redefines Table 3. published in [5].

The IND-CPA2 semantic security [10] is assured as members of the <g> set are indistinguishable from random tensors of the same size. More arguments and statistical evidence of tensor structures are provided [4].

5 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 1.29 s.

```
Print["............................................"]
Print["R-PROPPING OF A GROUP-BASED DIGITAL SIGNATURE"]
Print["............................................"]
dim = 3; Print["tensor dimension = ", dim];
period = 2^24 - 1; Print["tensor period = ", period];
g0 = {{158, 215, 6}, {216, 221, 53}, {45, 119, 206}};
r = RandomInteger[{2, period - 2}];
Print["random power = ", r];
Label[step1]; g = TFastPower[g0, r];
invg = TFastPower[g, period - 1];
If[TProd[g, invg] == IdentityMatrix[dim], nil,
  GoTo[step1]];
Print["random private generator  = ", MatrixForm[g]];
a = RandomInteger[{2, period - 2}];
Print["signer private key = ", a];
s0 = TFastPower[g, a];
Print["signer public key = ", MatrixForm[s0]];
k = RandomInteger[{2, period - 2}];
Print["signer session key = ", k];
Print["SIGNING PROCEDURE (s0,s1,s2)................."]
h = RandomInteger[{2, period - 2}];
Print["original message hashing= ", h];
kh = k h; Print["exponent k.h = ", kh];
s1 = TProd[ s0, TFastPower[invg, kh]];
Print["signature s1 = ", MatrixForm[s1]];
s2 = TFastPower[g, k];
Print["signature s2 = ", MatrixForm[s2]];
Print["VERIFYING PROCEDURE.........................."]
Print["recalculated message hashing = ", h];
s3 = TFastPower[s2, h];
Print["s3 = s2^h = ", MatrixForm[s3]];
Print["s1.s3=s0 ?  ", TProd[s1, s3] == s0]
```

Table 3. Small example program of the defined protocol. In a real-world application, dim =7 or greater should be used to get reasonable security.

....

R-PROPPING OF A GROUP-BASED DIGITAL SIGNATURE

....

tensor dimension = 3

tensor period = 16 777 215

random power = 15 410 182

random private generator = $\begin{pmatrix} 69 & 102 & 164 \\ 238 & 25 & 140 \\ 17 & 158 & 135 \end{pmatrix}$

signer private key = 13 481 815

signer public key = $\begin{pmatrix} 246 & 252 & 66 \\ 16 & 151 & 169 \\ 103 & 158 & 65 \end{pmatrix}$

signer session key = 6 686 110

SIGNING PROCEDURE (s0,s1,s2).................

original message hashing= 5 011 236

exponent k.h = 33 505 675 131 960

signature s1 = $\begin{pmatrix} 239 & 146 & 169 \\ 72 & 220 & 189 \\ 50 & 122 & 179 \end{pmatrix}$

signature s2 = $\begin{pmatrix} 82 & 181 & 131 \\ 150 & 206 & 21 \\ 253 & 63 & 28 \end{pmatrix}$

VERIFYING PROCEDURE..........................

recalculated message hashing = 5 011 236

s3 = s2^h = $\begin{pmatrix} 206 & 38 & 170 \\ 253 & 193 & 19 \\ 43 & 238 & 1 \end{pmatrix}$

s1.s3=s0 ? True

Table 4. The output of the sample program that was described in Table3.

6 Conclusions

We present a PQC class of a new digital signature based on group theory. The protocol is somehow resemblant to ElGamal's digital signature. 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 [11].

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, PQC: R-Propping of Public-Key Cryptosystems Using Polynomials over Non-commutative Algebraic Extension Rings, https://eprint.iacr.org/2020/1102, 2020
4. P. Hecht, R-Propping of HK17: Upgrade for a Detached Proposal of NIST PQC First Round Survey, https://eprint.iacr.org/2020/1217, 2020
5. P. Hecht, PQC: R-Propping of Burmester-Desmedt Conference Key Distribution, https://eprint.iacr.org/2021/024, 2021
6. 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
7. V. Roman'kov, Cryptanalysis of a combinatorial public key crypto-system, DeGruyter, Groups Complex. Cryptology. 2017.
8. A. Myasnikov, V. Shpilrain, A. Ushakov, Non-commutative Cryptography and Complexity of Group-theoretic Problems, Mathematical Surveys and Monographs, AMS Volume 177, 2011

9. FIPS PUB 197: the official AES standard, http://csrc.nist.gov/publications/fips/fips197/fips-197.pdf
10. 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
11. https://arxiv.org/a/hecht_p_1.html