Cryptanalysis of the Cryptosystems Based on the Generalized Hidden Discrete Logarithm Problem

Yanlong Ma

December 30, 2021

Abstract

The hidden discrete logarithm problem(HDLP) over non-commutative associative algebras (FNAAs) in [1] was broken in [8] by reducing to discrete logarithm problem(DLP) in a finite field through analyzing the eigenvalues of the representation matrix. A generalized form of HDLP(GHDLP) was proposed in [11], which is claimed to be computationally hard under quantum computers. Based on this, several schemes are proposed. In this paper, we will show that GHDLP can also be reduced to DLP in a finite field by algebraic representation. With all the instruments in hand, we will show how some schemes based on GHDLP can be broken. Thus we conclude that these schemes are not secure under quantum attack. So constructing schemes based on GHDLP is fundamentally wrong.

1 Introduction

We first recall the integer factorization problem(IFP) and the discrete logarithm problem(DLP):

IFP: given a big number of the form $n = pq$, find the two prime divisors p and q.

DLP: having known g, h in a cyclic group G, find $t \in N$ such that $g^t = h$.

IFP and DLP have been used as mathematical base of cryptography for a long time. RSA and ElGamal may be the most famous two. Until now, a lot of digital signatures and cryptosystems are still using the two difficult problems. They are proved by time to be usable for a long time since there is still no polynomial methods to break them in classical computers.

However, quantum computers can solve the two difficult problems in very short time.[6] So new difficult mathematical problems are in urgent need. In this background, a great many problems are proposed and announced to be safe under quantum computers. A possible try is to construct equations with several variables in stead of one, for example, Rainbow in [7] is built based on the difficulty of solving multivariable polynomial systems. The hidden discrete logarithm problem(HDLP) proposed in [1] is also one try to extend the one variable problem DLP to an equation of several variables. The units(or just local units) are used to hide the initial element. This is similar in spirits to the methods in [23, 36, 37] and [24, 35]. in [23] Also, in [32, 33, 34], matrix are used in multivariate schemes to diffuse the initial functions or change the basis to hide the initial one.

HDLP is defined in a finite non-commutative associative algebra (FNAA). Let V be a vector field of dimension m over the finite field $GF(q)$. By an operation on V we just mean a map $f : V \times V \to V$, while instead of $f(s, t) = u$ we often write $sft = u$. Endowed V with a bilinear operation which we call multiplication and denote it by a round dot or just omit it. Then for each $f_i \in GF(q), v_i \in V$ we have:

$(f_1v_1 + f_2v_2) \cdot v_3 = f_1v_1 \cdot v_3 + f_2v_2 \cdot v_3$, $v_3 \cdot (f_1v_1 + f_2v_2) = f_1v_3 \cdot v_1 + f_2v_3 \cdot v_2$

Then V together with the multiplication is called an algebra. We may still call the underlying space V an algebra, when the multiplication is known. If in particular the operation is not commutative but associative, V is called an FNAA.

Now we can define HDLP [1, 17] : Suppose A is an FNAA, $B \subset A$ is a given subspace. Given two elements x, y in A, find a unit (invertible element) $u \in B$, and an integer t, such that $ux^tu^{-1} = y$, if they exist. The answer may not be unique, but any one may be okay since they will give an equivalent key in the schemes. But in most cases, t is unique in $Z/o(x)$, where $o(x)$ is the order or local order of x. This is to say, if (u, t) and (u', t') are solutions to $ux^tu^{-1} = y$, then $x^t = x^{t'}$.

In [11, 18], a more general form of HDLP(GHDLP) is proposed: Given two elements x, y in A, compute a triple (t, u, v), such that $ux^t v = y$, and $r = vu$ is a global (or local) right unit: $ar = a$, for all $a \in A$ (or just for $a = x$, respectively). Still, the solutions may not be unique, u, v may be required to lie in a given subspace B.

In the definition above, "right" can be replaced with "left". Then a similar problem is proposed, which is still called GHDLP.

Based on HDLP, one can construct an interesting method to exchange secrets(A Diffie-Hellman, in more sophisticated voice)[1]: Publicly choose a big prime number p, an FNAA A of dimension m over $GF(p)$, a big subalgebra B of A, and a element $x \notin A$. Now (p, A, m, B, x) are known to all people.

To exchange secrets, Alice choose secretly a unit $g \in B$ together with a secret integer number t while Bob choose secretly another unit $h \in B$ and integer number s. Now, only Alice knows (g, t) and only Bob knows (h, s). Then Alice compute $k_1 = gx^t g^{-1}$ and send it to Bob. Bob compute $k_2 = hx^s h^{-1}$ and send it to Alice. Now Alice knows (g, t, k_2) and computes $k_A = gk_2^t g^{-1} = ghx^{st} h^{-1} g^{-1}$. Bob knows (h, s, k_1) and computes $k_B = hk_1^s h^{-1} = hgx^{ts} g^{-1} h^{-1}$. Now since g and h are chosen in a commutative subalgebra B of A, $gh = hg$, thus $k_A = k_B$ and they share a common secret $k = k_A = k_B$.

In [1], a digital signature is constructed similarly using a publicly known hash function H. (p, A, m) are also publicly known. Now if Alice wants to sign something, she secretly choose a number t, $x, h, g, v \in A$, such that x, g, h do not commute with one another, x is only invertible in some subalgebra B of A with a local unit v. This is to say, $x \in B$ and there exist some $x' \in B$ such that $xx' = x'x = v$, with $vb = bv = b$, for all $b \in B$. Now (z, y, w) where $z = hxh^{-1}, y = gx^t g^{-1}, w = gvh^{-1}$ are published as public keys, (x, h, g, t) are kept as secret keys. Here secret keys are used to sign a massage. They represents the identification of the signer so they must be kept secretly. Public key are used to verify a signature, which can be done by any one who receives Alice's message, so any one should know it when needed.

Now if Alice wants to sign a message M, she will first randomly choose k, compute $u = gx^k h^{-1}, e = H(M, u)$, and then sign M with e and $s = k - te \bmod o(x)$ where $o(x)$ is the order of x.

Suppose Bob get (M, e, s) from Alice, He can verify it in the following procedure. He will compute $u' = y^e w z^s, e' = H(M, u')$ and then check whether $e = e'$. If the signature is valid, then $u' = y^e w z^s = gx^{te} g^{-1} gvh^{-1} hx^s h^{-1} = gx^k h^{-1} = u$. Here we have used that $s + te = k$ and that $xv = vx = x$ because v is a local unit to the subalgebra containing x.

However, these two problems are not post-quantum safe, and even not classical safe when then size of the keys are too small.[5] gives an attack towards HDLP in special cases where x has a nonzero and non-identity 'determinant'. This can be resisted by taken non-special elements. [8] gives a classic attack towards HDLP with time $O(t^{1/2})$, where t is the order of the base element x. The invertible element u is also computed. But there is a small mistake there. We will correct it and demonstrate it again in a more clear way. We will also improve the estimation about the complexity, by citing a stronger method of computing DLP in finite fields.

If quantum computer is available, then a polynomial attack to GHDLP is possible. In [9], the author gives a quantum attack towards two concrete signatures in [10] using HSP(Hidden subgroup problem). It also works for the signatures in [4], [14] and [18]. In this paper, we will break a family of related systems by reducing them to DLP in finite fields, which is of polynomial time using quantum computers[6]. Thus constructing schemes based on HDLP is fundamentally wrong.

These attacks work just because using conjugation in HDLP or quasi-conjugation in GHDLP can not really hide all information about the exponentiation. Linear algebra tells us that such operations do not change the determinant and at least one eigenvalue. Then one have an important observation: eigenvalues of a power of a matrix are just the powers of eigenvalues at first. These two propositions from linear algebra will help us work out all we need. A similar analysis, but for different schemes, can be found in [20] and [21]. In [5] , the determinant homomorphism attack(DHA) is explained in details. If the attack above is impossible, then one can try eigenvalue attack(EA). In this case, we may extend the field if necessary. All the attacks above requires that we must transform the problems to a matrix form. Since FNNA are often defined using a group or just a semigroup, any representation of the initial (semi)group will extend to a representation of the FNNA. One can also using regular representations as well. To demonstrate this, a little algebra and representation theory is shown. Once GHDLP is solved, all the schemes based on it are broken.

2 How to break the two schemes if HDLP is solved

Suppose now we can solve HDLP in any FNNA. We will show the two schemes referred in the introduction is not safe.

2.1 How to break the Diffie-Hellman

Suppose Carol wants to obtain Alice and Bob's common secret. By listening to the internet, he can obtain $(p, A, m, B, x, k_1, k_2)$. Now he can solve the HDLP of $k_1 = gx^t g^{-1}$, and then he will know a pair t', g' with $g' \in B$, such that $k_1 = g' x'^t g'^{-1}$. Carol can calculate h' and s' in the same way. Now Carol can compute the secret $g'h'x^{s't'}h'^{-1}g'^{-1} = ghx^{st}h^{-1}g^{-1}$, as one can easily check. Thus the Diffie-Hellman is broken. The schemes in [19] and [20] can be similarly broken as well.

2.2 How to break the digital signature

Suppose now Carol wants to forge Alice to sign messages. He knows $z = hxh^{-1}, y = gx^t g^{-1}, w = gvh^{-1}$ for some h, x, t, g, v. So if $d = gh^{-1}$, then $y = dz^t d^{-1}$. By solving HDLP, Carol now knows (t', d'), such that $y = d' z^{t'} d'^{-1}$. Then Carol can compute $wz^{k'} = gvh^{-1}hx^{k'}h^{-1} = gx^{k'}h^{-1}$, where k' is randomly chosen. Furthermore, he can compute $e = H(M, u), s = k' - t'e$ and thus he can sign as if he was Alice.

3 About other systems

In this section, suppose we can solve GHDLP.

3.1 About the zero-knowledge protocol in [19]

In [19], a zero-knowledge protocol based on the general form of HDLP is constructed: Now Bob wants to know if Alice knows the private key (t, x) corresponding to the public key y, where $y = b^t g^x a^t$, where g is locally invertible, a, b, g is known to all, $ab = r$ is a local unit to g. The author constructed a series of procedures to verify that if Alice knows. But as we have shown, this do not make sense when GHDLP is solved, because any one who knows y can computes a corresponding x, t. Similar arguments goes to the cryptosystems in [20].

3.2 How to break the digital signature in [18]

In [2], a signature scheme is also proposed: Alice choose two commutative elements N, G of prime order q in A, then she secretly choose two integers t, w, and compute $J = N^t U^w$, then she choose secretly $A_1, B_1, A_2, B_2, B_3, A_4, B_4$, such that $R_1 = B_1A_1, R_2 = B_2A_2, R_3 = B_3A_1, R_4 = B_4A_4$ are all global right-sided element. Next, she will compute $Z_1 = A_1NUB_1, Y_1 = A_2N^xB_2, T_1 = RA_1B_2, Z_2 = A_1JU^2B_3, Y_2 = A_4J^{x/2}B_4, T_2 = R'A_1B_4$, where R, R' are secret global right-sided unit. Then $(Z_1, Y_1, Z_1, Z_2, Y_2, Z_2)$ is published as a public key.

To sign a message M, generate a random $K \in A$,compute $V_1 = KN^kB_2, V_2 = J^{k/2}B_4, e = H(M, V_1, V_2), s = k + xe \ mod \ q$. Alice then calculate a solution S of the following equation:$SA_1U^s = K$, and finally the signed message is (M, e, s, S). To verify the signature, Bob will compute $V_1' = SZ_1^sT_1Y_1^{-e}, V_2' = SZ_2^{s/2}T_2Y_2^{-e}, e' = H(M, V_1', V_2')$ and finally check if $(M, e, s, S = (M, e', s', S)$.

To break this signature, we may first observe that it is also unnecessary to know the real secret key, an equivalent one is just okay. we first find x and P, Q. In attacking this signature, the right regular representation R in section 5.2 is used. This is to map all right-sided units to the identity matrix. So under the representation, the equations about the public keys become

$$\begin{cases} Z_1 = O_1NUO_1^{-1}, Y_1 = O_2N^xO_2^{-1}, T_1 = O_1O_2^{-1} \\ Z_2 = O_1JU^2O_1^{-1}, Y_2 = O_4J^{x/2}O_4^{-1}, T_2 = O_1O_4^{-1} \end{cases} \tag{1}$$

So if we take $P = O_1 N O_1^{-1}, Q = O_1 U O_1^{-1}, H = O_1 J O_1^{-1}$, then $PQ = QP$, and they both have order q. Now the equations above can be translated to the following one :

$$\begin{cases} Z_1 = PQ, L_1 = T_1 Y_1 T_1^{-1} = N^x \\ Z_2 = HQ^2 = P^t Q^{w+2}, L_2 = T_2 Y_2 T_2^{-1} = P^{tx/2} Q^{wx/2} \end{cases} \tag{2}$$

So $Z_2^{x/2} = P^{tx/2} Q^{wx/2+x} = L_2 Q^x, Z_1^x = P^x Q^x$ and thus $L_1 L_2^{-1} = (Z_1^2 Z_2^{-1})^(x/2)$ so x can be computed. Once x is known, $P = L_1^y$ is known, where y is the inverse of x in Z_q^*. So $Q = P^{-1} Z_1$ and $H = Z_2 Q^{-2}$ are also known.

Now, go back to the algebra, and we get the preimages of P, Q, H. We may still denote them by P, Q, H. Use these data, we can sign any message as if we are Alice now: First we select a random K and compute $V_1 = K P^k T_1, V_2 = K H^{k/2} T_2, e = H(M, V_1, V_2), s = k + xe, SQ^s = K$. One can easily check that this satisfies the verification, and thus the signature is broken. The signature in [26] and [27] can be broken similarly.

3.3 How to break the digital signature in [2]

In [2], the author gives another signature scheme:

Alice selects two commutative elements $g, h \in A$, two unit $a, b \in A$, and two integer numbers x, w. These six form the private keys. Then she compute $u = a g^x h b^{-1}, y = b g b^{-1}, z = b h^w a^{-1}$, and announce (u, y, z) as public keys.

To sign a message m, Alice selects two random integers k, t and computes $v = a g^k h^t a^{-1}, e = H(m, v)$, then she solve a system of equations in $GF(q)$ to get two numbers s and l:

$$\begin{cases} es^2 + xs + xl = k \\ s + ws + l + wl = t \end{cases} \tag{3}$$

Then the signed message is (m, e, s, l).

Computing $v' = (u y^{es} z)^s (uz)^l$ and checking if $H(m, v')$ is equal to e verifies this signature.

How can we imitate Alice? Now we knows u, y, z and can compute $uz = a g^x h b^{-1} b h^w a^{-1} = a g^x h^{w+1} a^{-1}$, $zu = b h^w a^{-1} a g^x h b^{-1} = b g^x h^{w+1} b^{-1}$. So we know $uz = rzur^{-1}$, where $r = ab^{-1}$. Using the method in section 4.3, we can find what r is. Then if we set $f = bhb^{-1}$, then $zr = b h^w b^{-1} = f^w$,$r^{-1} u f^{-1} = y^x$. Take $w' = 1$, that is we take $f = zr$ and compute x' in the DLP $r^{-1} u f^{-1} = y^{x'}$, then we get an equivalent secret key (w', x'). Now we can imitate Alice to sign m:

$v = a g^k h^t a^{-1} = a b^{-1} y^k f^t b a^{-1} = r y^k f^t r^{-1}, e = H(m, v)$. s and l can be computed in the same way as Alice because we now know an equivalent private key (x', w').

Signatures in [15] can be broken in a similar way.

3.4 How to break the digital signature in [11]

In [11], a signature scheme is presented as well: Alice firstly selects a non-invertible element g in A together with a local left-sided unit l and local right-sided unit r. Then she compute $g' = lg, y' = r g^x$. The private key is (l, r, x, g), the public key is g', y'.

To sign a message m, Alice choose a random integer k, and compute $u = r g^k l, v = H(m, u), s = k - xv$, where the last equation is in $GF(q)$. Then the signed message is (m, v, s).

This signature is verified in the following procedures:

1.Compute $u' = y'^v g'^s, v' = H(m, u')$.

2. Check if $v = v'$.

So how can we break this schemes?

Suppose the local inverse of l, r is l^{-1}, r^{-1}, then l^{-1}, r^{-1} must be local one-sided unit, too. So $g' = gl = l^{-1} g l, y' = r g^x = r g^x r^{-1}$, So $y' = w g'^x w^{-1}$ for some $w = rl$, which is also local invertible. One can see that a conjugation by local invertible element can also pass the procedures we demonstrate in the following sections. Thus x and w is computed using GHDLP, and now we know (g', y', x, w) So we can imitate Alice to sign a message m as follows:

choose a random integer k, and compute $u = r g^k l = r l g'^k r l = w g^k w, v = H(m, u), s = k - xv$. So the signature can be forged.

Signatures in [12], [13] and [14] can be broken in a similar way. Other schemes constructed based on GHDLP on different algebras(as in [16] and [25]) can also be broken in the same way, because our algorithm is independent of the algebra itself. Changing algebras just changing the representing matrix, and thus cannot really resist our attack.

4 GHDLP in matrix form

Now everything is over if GHDLP is solved. In this section, We first solve GHDLP in matrix algebra. For GHDLP in any FNAA, one need more algebraic preliminaries. We will treat those later. However, all the spirits are demonstrated here.

4.1 Solution to HDLP in matrix algebra

We first give the solution to HDLP. The problem in matrix form is: Given matrix X, Y of order m, find $n \in N, U \in GL(m, F)$ such that $U^{-1}X^nU = Y$, where F is the finite field we are considering.

Extend the field if necessary, X can be changed to its Jordan form. The extension field can be chosen as $K = F[x]/f(x)$, where $f(x) = det(xI - X)$ is the characteristic function of X. This can be computed of time $O(m^3log^2p)$ using Gaussian elimination, where log^2p is the time cost of multiply two elements in GF(p). f can also be chosen to be the minimal polynomial of X when convenient, which will decrease the size of the extended field K.

By linear algebra we know $U^{-1}X^nU$ have the same characteristic polynomial with X^n, the eigenvalues of X^n consist of nth power of the eigenvalues of X. The roots of the characteristic polynomial are just the eigenvalues. Comparing the eigenvalues on both sides of the equation $U^{-1}X^nU = Y$, we will get an equation of sets for which repetitions are allowed: $\{\lambda_1^n, \lambda_2^n, ..., \lambda_{n-1}^n, \lambda_m^n\} = \{\tau_1, \tau_2, ..., \tau_{n-1}, \tau_m\}$, where $\{\lambda_1, \lambda_2, ..., \lambda_{n-1}, \lambda_m\}$ are eigenvalues of X with multiplicity, and $\{\tau_1, \tau_2, ..., \tau_{n-1}, \tau_m\}$ are are eigenvalues of Y with multiplicity. To compute these eigenvalues, one need compute roots of polynomial in a finite field, which cost $O(d^3log(q))$ if we use the probabilistic Cantor–Zassenhaus algorithm.[30] Here q is the order of the extended field $K = GF(q)$, d is the degree of the polynomial. This is polynomial in both $log(q)$ and $d = m$. When GRH(generalized Riemann Hypothesis) is assumed to be true, a deterministic polynomial algorithm can be found in [30].

Then at most m (linear in m) matches are needed before we can find n: we may first select an element (one may select at most m times) in $\{\lambda_1, \lambda_2, ..., \lambda_{n-1}, \lambda_m\}$, say λ_i, and solve DLP at most m^2 times for each τ_j with base λ_i, one of these discrete log must be n. If we have factorized the polynomials in $GF(p)$, the procedure goes faster, we can just match a subset, not all of the m elements, using the knowledge of degree and multiplicities. In [8], the author omits the matches. This is wrong, because the first r-degree root of the characteristic polynomial may match another r-degree root.

All together, we find a polynomial method to reduce the problem to at most m^2 DLPs. In most cases, just $O(m)$ DLPs is okay.

When n is found, it is easy to find one $U \in A$ by using Gaussian elimination to a linear fuction $X^nU = UY$ in m^2 variables (the entries of U) and m^2 equations. The total cost here is about $O(m^6log^2q)$. Taking an intersection with B gives a correct one element. Also, in practice, it is more efficient to compute such a U in equation of the FNNA, and the total cost becomes $O(m^3log^2q)$, since an FNNA of dimension m has only m components.

Since DLP in finite fields can be done in polynomial time by [6], we can solve HDLP in polynomial time. By the way, we know that just expanding m, the size of the matrix, do not lift the security, because this just increase the difficulty of the matches of eigenvalues in polynomial coefficients (no more than m^2).

4.2 Solution to GHDLP in matrix algebra

In matrix form, GHDLP is: Given X and Y, find $U, V \in B, t \in N$, with $VU = R, XR = X, Y = UX^tV$. The GHDLP can also defined demanding $RX = X$, or they both hold. In practice, X comes from a representation and thus almost always have a special Jordan form: For some invertible S, $SXS^{-1} = \begin{pmatrix} A & O \\ O & B \end{pmatrix}$, with A digonal and B Jordan blocks of eigenvalues 0.

(This can be proved true if the local order r of X do not divide the field character p: Since $X^{2r} - X^r = 0$, we know the minimal polynomial $f(x)$ of X divides $x^{2r} - x^r = 0$, which have no repeated root except a possible 0, this is because $x^r - 1$ has no repeated roots since its derivative rx^{r-1} is not zero when $x \neq 0$.)

We will prove that our methods still work for GHDLP in the above case. One can still compute the eigenvalues of X and match them to the eigenvalues of Y, then at least one in the m^2 DLPs will give the right t.

Suppose X has eigenvalues $\lambda_i \neq 0, 1$ with eigenvector $\vec{v}_i$(X always has such an eigenvalue, or it can not have many different powers and thus can not be used in crptosystems), then at least one of such eigenvalues is preserved in UXV: $UXVU\vec{v}_i = UX\vec{v}_i = U\lambda_i\vec{v}_i = \lambda_i U\vec{v}_i$. So either $U\vec{v}_i = \vec{0}$, or it is an eigenvector with the same eigenvalue λ_i. However, $U\vec{v}_i = \vec{0}$ can not always holds, since the eigenvalues $\vec{v}_i$ span the column space of X^l for $l \geq m$. So if $U\vec{v}_i = \vec{0}$ for all v_i, then $UX^lV = 0$, a contradiction. This shows that UX^tV and X^t have common eigenvalues, and thus our methods work.

For generalized form, the analysis above do not work, we will use another instrument in linear algebra, the root subspace decomposition. $V = \oplus N((X - \lambda_i I)^{r_i})$, where $r_i + 1$ is equal to the size of the Jordan block corresponding to λ_i, which may not be different from another λ_j. Then if $\vec{v}_i \in N((X - \lambda_i I)^{r_i})$, then $(X - \lambda_i I)^{r_i}\vec{v}_i = \sum_{j=0}^{r_i}(-\lambda_i)^{r_i-j}C_{r_i}^j X^j\vec{v}_i = 0$ And thus $(AXB - \lambda_i I)^{r_i}A\vec{v}_i = \sum_{j=0}^{r_i}(-\lambda_i)^{r_i-j}C_{r_i}^j AX^jBA\vec{v}_i = A(\sum_{j=0}^{r_i}(-\lambda_i)^{r_i-j}C_{r_i}^j X^j\vec{v}_i) = 0$ So either $A\vec{v}_i$ is zero, or it is a generalized eigenvector corresponding to λ_i. Now since the sum of subspace corresponding to nonzero eigenvalues form the column space of X^l for large l, there must be a common eigenvalue between X and AXB.

Now we try to find U and V. Now $Y = UX^tV$ goes to $YU = UX^t$, since $VU = R$ is a right unit to X. Then by taking intersection with B, we compute U out. Then V can be computed by solving $Y = UX^tV$ and take an intersection.

So far we have solve GHDLP in matrix forms.

5 HDLP and GHDLP in any FNNA

To solve HDLP and GHDLP in a general FNNA, we reduce it to the matrix case.Here we introduce some algebraic instruments, including structure constants and representation theory.

5.1 Structure constants

To describe multiplication in an FNNA A, we choose a basis(as a vector space) $B = \{e_1, ..., e_m\}$ of A, then if all the multiplications of any two elements of B is given:$e_i \cdot e_j = \sum_{k=1}^m \Gamma_{i,j}^k e_k$, then we can know all the multiplications of any two elements of V, just by the bi-linearity of the multiplication.

The coefficients of $e_i \cdot e_j$, say, $\Gamma_{i,j}^k$ is called the structure constants of A corresponding to the basis B. Clearly, for a given basis, the structure constants and the multiplication determine each other.

5.2 Algebraic representation

A representation of an associative algebra A is by definition an algebraic homomorphism ϕ from A to $End(W)$, the algebra of all linear transformations of W, with trivial addition and composition as multiplication.

Now we consider the left regular representation L, with $L(a) = L_a \in End(A)$, where $L_a(r) = a \cdot r$,for all $r \in A$). Respectively, we can also consider the right regular representation R,with $R(a) = R_a \in End(A)$, where $R_a(r) = r \cdot a$, for all $r \in A$). In most cases the left regular representation is enough, but sometimes the right regular representation is more convenient.

Then L can be easily proved to be an homomorphism: $(L(a)L(b))(r) = abr = L(ab)(r)$, so $L(a)L(b) = L(ab)$. The same arguments goes to R, the only difference is that R is an antihomomorphism: $R(ab) = R(b)R(a)$.

One can prove that if A is chosen with no nonzero left(right) annihilator, then L(R,respectively) is injective, since their nontrivial kernel consists of such elements. For algebras with no nonzero left annihilators, we may first project the algebra to its quotient A/N, where N is the annihilators, which must be a left ideal and of course a subspace. For the right annihilators, the same method works as well. In these cases, the dimension of the algebra is reduced, and things become easier.

So any associative algebra and in particular any such FNAA can be considered as a subalgebra of an algebra of the form $End(V)$ when divided by their annihilators, where V is a vector space.

Besides the left and right representation, other presentations can also be used, when it is convenient or more natural. For example, when an FNAA is constructed from a group, then the irreducible representations can always extend to the FNAA, which is often of less dimension than the regular representations.

5.3 Representation described as structure constants

Now suppose we are given an algebra A with a basis B, together with the structure constants $\{\Gamma_{i,j}^k\}$. we will determine explicitly the representation, using matrix language.

For any vector $v \in A$, we have $v = \sum_{i=1}^m v^i e_i$, for some $v^i \in GF(p)$, then $L_v(e_j) = L_{\sum_{i=1}^m v^i e_i}(e_j) = (\sum_{i=1}^m v^i e_i) \cdot e_j = \sum_{i=1}^m v^i e_i \cdot e_j = \sum_{i=1}^m v^i (\sum_{k=1}^m \Gamma_{i,j}^k e_k) = \sum_{k=1}^m (\sum_{i=1}^m v^i \Gamma_{i,j}^k) e_k$

Let $c_j^k = \sum_{i=1}^m v^i \Gamma_{i,j}^k$, then we have $L_v(e_j) = \sum_{k=1}^m c_j^k e_k$. So the matrix of L_v is $\{c_j^k\}$,that is, c_j^k lies on the jth row crossing the kth column.

If we identify $(v^1, ...v^m)$ with v, and rename L as ϕ, then we see the injective homomorphism: $\phi(v^1, ...v^m) \mapsto \{\sum_{i=1}^m v^i \Gamma_{i,j}^k\}$.

5.4 Reduction of HDLP and GHDLP to matrix algebra

In the above subsection, we have shown that any FNAA can be treated as a subalgebra of some matrix algebra. Since we can do HDLP in matrix algebra, we can do it in any FNAA: With a ϕ action to both side of $ux^t u^(- 1) = y$, one can get $\phi(u)(\phi(x))^t(\phi(u))^{-1} = \phi(y)$. This can be calculate then because this is HDLP in matrix algebra. Solve this we get t, and the same solution goes to the initial equation because ϕ is injective. For GHDLP, the equation is $\phi(u)(\phi(x))^t(\phi(v)) = \phi(y)$, with $xvu = x$.

Now all the reduction above can be done in polynomial time even in classical computer, so HDLP and GHDLP is never safe in any FNNA.

6 Some examples of matrix representations

In all of the reference[1, 2, 3, 4, 5] about FNAAs, given i, j, $\Gamma_{i,j}^k \neq 0$ for just one k, this makes the matrix much simpler and one can even construct a basis-vector multiplication table (BVMT). For example, about Table 1 in [1], we can construct the map ϕ:((a, b, c, d) means $a\vec{e} + b\vec{i} + c\vec{j} + d\vec{k}$)

$$\phi(a, b, c, d) = \begin{pmatrix} \mu a & -\mu^{-1}\tau b & -\mu^{-1}c & -\mu^{-1}\tau d \\ \mu b & \mu a & -d & c \\ \mu c & \tau d & \mu a & -\tau b \\ \mu d & -c & b & \mu a \end{pmatrix} \tag{4}$$

Table 1: Table 1: The basis-vector multiplication table

$\cdot$	$\vec{e}$	$\vec{i}$	$\vec{j}$	$\vec{k}$
$\vec{e}$	$\mu\vec{e}$	$\mu\vec{i}$	$\mu\vec{j}$	$\mu\vec{k}$
$\vec{i}$	$\mu\vec{i}$	$-\mu^{-1}\tau\vec{e}$	$\vec{k}$	$-\tau\vec{j}$
$\vec{j}$	$\mu\vec{j}$	$-\vec{k}$	$-\mu^{-1}\vec{e}$	$\vec{i}$
$\vec{k}$	$\mu\vec{k}$	$\tau\vec{j}$	$-\vec{i}$	$-\mu^{-1}\tau\vec{e}$

As for Table 2(Table 1 in [4]), we can also construct the corresponding map ϕ:((a, b, c, d) means $a\vec{e_0} + b\vec{e_1} + c\vec{e_2} + d\vec{e_3}$)

$$\phi(a, b, c, d) = \begin{pmatrix} a + \tau d & 0 & a + d & 0 \\ 0 & b + c & 0 & \tau b + c \\ \tau b + c & 0 & b + c & 0 \\ 0 & a + d & 0 & a + \tau d \end{pmatrix} \tag{5}$$

For Table 3(Table 2 in [4]), we also construct another corresponding map ϕ:((a, b, c, d, e, f) refers $a\vec{e_0} + b\vec{e_1} + c\vec{e_2} + d\vec{e_3} + e\vec{e_4} + f\vec{e_5}$)

Table 2: Table 2:BVMT2

$\cdot$	$\vec{e}_0$	$\vec{e}_1$	$\vec{e}_2$	$\vec{e}_3$
$\vec{e}_0$	$\vec{e}_0$	$\vec{e}_3$	$\vec{e}_0$	$\vec{e}_3$
$\vec{e}_1$	$\tau\vec{e}_2$	$\vec{e}_1$	$\vec{e}_2$	$\tau\vec{e}_1$
$\vec{e}_2$	$\vec{e}_2$	$\vec{e}_1$	$\vec{e}_2$	$\vec{e}_1$
$\vec{e}_3$	$\tau\vec{e}_0$	$\vec{e}_3$	$\vec{e}_0$	$\tau\vec{e}_3$

$$\phi(a,b,c,d,e,f) = \begin{pmatrix} a+\lambda f & 0 & \tau a+\mu f & 0 & a+f & 0 \\ 0 & b+e & 0 & \mu b+\tau e & 0 & \lambda b+e \\ c+\lambda d & 0 & \tau c+\mu d & 0 & c+d & 0 \\ 0 & c+d & 0 & \tau c+\mu d & 0 & c+\lambda d \\ \lambda b+e & 0 & \mu b+\tau e & 0 & b+e & 0 \\ 0 & a+f & 0 & \tau a+\mu f & 0 & a+\lambda f \end{pmatrix} \tag{6}$$

Table 3: Table 3:BVMT3

$\cdot$	$\vec{e}_0$	$\vec{e}_1$	$\vec{e}_2$	$\vec{e}_3$	$\vec{e}_4$	$\vec{e}_5$
$\vec{e}_0$	$\vec{e}_0$	$\vec{e}_5$	$\tau\vec{e}_0$	$\tau\vec{e}_5$	$\vec{e}_0$	$\vec{e}_5$
$\vec{e}_1$	$\lambda\vec{e}_4$	$\vec{e}_1$	$\mu\vec{e}_4$	$\mu\vec{e}_1$	$\vec{e}_4$	$\lambda\vec{e}_1$
$\vec{e}_2$	$\vec{e}_2$	$\vec{e}_3$	$\tau\vec{e}_2$	$\tau\vec{e}_3$	$\vec{e}_2$	$\vec{e}_3$
$\vec{e}_3$	$\lambda\vec{e}_2$	$\vec{e}_3$	$\mu\vec{e}_2$	$\mu\vec{e}_3$	$\vec{e}_2$	$\lambda\vec{e}_3$
$\vec{e}_4$	$\vec{e}_4$	$\vec{e}_1$	$\tau\vec{e}_4$	$\tau\vec{e}_1$	$\vec{e}_4$	$\vec{e}_1$
$\vec{e}_5$	$\lambda\vec{e}_0$	$\vec{e}_5$	$\mu\vec{e}_0$	$\mu\vec{e}_5$	$\vec{e}_0$	$\lambda\vec{e}_5$

Thus we can see that whatever the FNAAs seems to be complicated, it behaves no more than the corresponding matrix algebra.

7 Concrete examples

In this section, we use brackets to denote codes. Only key codes are shown here. All the codes in the following can be run within one second in the calculator (it uses the machine V2.26-9) at the official website of Magma:(see http://magma.maths.usyd.edu.au/calc/).

7.1 A concrete example of solving HDLP

In this subsection, We use the FNAA with BVMT given in Table 1 in section 6.

The parameters are :$\mu = 1, \tau = 1, p = 823057273411232437763981$(this is a number of about 80 bits, as the author gives in [5]). This p is found using Magma by a random integer less than $2^{80} = 1208925819614629174706176$ composed by a function which finds the first prime behind a given number:

[NextPrime(Random(1208925819614629174706176));]

Now A is the quaternion-like algebra(The only difference is that this is in a finite field.) And so the following representation is more natural from algebra:

$$\phi(a,b,c,d) = \begin{pmatrix} a & d & -c & b \\ -d & a & -b & -c \\ c & b & a & -d \\ -b & c & d & a \end{pmatrix} \tag{7}$$

One can easily get the components of a vector from the entries of its representing matrix. Then we set

$$x = (401061284016651457375229, 140498875087816015438562, \\ 407926716616598678661430, 661420587423165019065860)$$

$$u = (627968683547310283890579, 760824309936500718069644, \\ 523867729517403306136890, 301803738536395935875170)$$

The components in this two vectors are generated in Magma using random integers less than p, 8 times altogether.

Then we can compute:

$$u^{-1} = (42602107029244693043196 9, 52159394946311503065955 8, \\ 47304470195395368356896 5, 66135365077432236155589 7)$$

we can then compute a random number 46548060726748433200355 as the power parameter, this is the main secret n. We use it to compute the hidden power:

$$y = u \cdot x^{46548060726748433200355} \cdot u^{-1} = (79423095610761169388559 7, 28867191652466967492364 4, \\ 19799583312762069352466 5, 50359205096408517009191) =: (e, f, g, h)$$

Take (n, u) as secrets, and publicly show others x and y.

Now we compute n and u from x and y.

First, we represent x and y in matrix with X and Y through the representation ϕ mentioned above. Then we have an equation in matrix algebra:

$$UX^nU^{-1} = Y \tag{8}$$

We then try to compute the Jordan form of X and Y (This can give all the eigenvalue quickly using computer), see details in the appendix.

[JordanForm(X);JordanForm(Y);]

Then all the eigenvalues of X and Y is known.

The eigenvalues of X are 549650594809927781389712,252471973223375133360746.

The eigenvalues of Y are 613363570925274286294157,152041067878716663713056.

We may first try the determinant attack: Try to find the discrete log of Y in the base of X:

[Log(Determinant(X),Determinant(Y));]

This works, and gives

[46548060726748433200355].

In special cases, the determinants of both X and Y are 1 or 0, but our method still works, because the eigenvalues can not always be 1 or 0. One may try to compute the dicrete logs of the eigenvalues. One can see the codes and outputs in the appendix.

Then we may compute u. suppose $u = (s, t, u, v)$ now they can be calculated out by solving the linear system $ux^n = yu$. One can first compute

$z = x^n = (79423095610761169388559 7, 82086655627072113909819 9,$
$16352279598459885965586 2, 74258274159242819107072 1) = (E, F, G, H)$

Then to find s, t, u, v we refer to the equation $(s, t, u, v)(E, F, G, H) = (e, f, g, h)(s, t, u, v)$, which is equivalent to the following system:

$$\begin{cases} Es - Ft - Gu - Hv = es - ft - gu - hv; \\ Fs + Et + Hu - Gv = fs + et - hu + gv; \\ Gs - Ht + Eu + Fv = gs + ht + eu - fv; \\ Hs + Gt - Fu + Ev = hs - gt + fu + ev; \end{cases} \tag{9}$$

This is just

$$\begin{cases} (E - e)s - (F - f)t - (G - g)u - (H - h)v = 0; \\ (F - f)s + (E - e)t + (H + h)u - (G + g)v = 0; \\ (G - g)s - (H + h)t + (E - e)u + (F + f)v = 0; \\ (H - h)s + (G + g)t - (F + f)u + (E - e)v = 0; \end{cases} \tag{10}$$

Then vectors in the null space of the coefficient matrix gives what we need:

(1 0 29468210829750626781865 3 17886283966910151881866)

One may find this is not the u at first, this does no harm, because it is an equivalent one that gives the same conjugation. In practice, it matters only that we know such a u that satisfies the equation, it is unnecessary to find the one generated secretly.

7.2 A concrete example of solving GHDLP

In this subsection, We use the FNAA with BVMT given in Table 2 in section 6.

We use the parameters $\tau = -1, p = 823057273411232437763981$

$$x = (401061284016651457375229, 140498875087816015438562,\\ 140498875087816015438562, 401061284016651457375229)$$

$$u = (26131572261834006134192, 21391477283755189959 0870,\\ 21391477283755189959 0870, 26131572261834006134192)$$

$$v = (627968683547310283890579, 760824309936500718069644,\\ 178256814794268587889944, 627968683547310283890579)$$

Then one can check $xvu = x$. The new representation is

$$\phi(a,b,c,d) = \begin{pmatrix} a-d & 0 & a+d & 0 \\ 0 & b+c & 0 & -b+c \\ -b+c & 0 & b+c & 0 \\ 0 & a+d & 0 & a-d \end{pmatrix} \tag{11}$$

So the matix form of X is

$$\begin{pmatrix} 0 & 0 & 9781172448522593528 0767 & 0 \\ 0 & 46227468992939704387227 & 0 & 0 \\ 0 & 0 & 46227468992939704387227 & 0 \\ 0 & 9781172448522593528 0767 & 0 & 0 \end{pmatrix}$$

We still use the number 46548060726748433200355 as the power parameter, this is the main secret n. We use it to compute the hidden power:

$$y = u \cdot x^{46548060726748433200355} \cdot v = (24417931581393991043416, 73030926097725151815655,\\ 16173935508496717467 4242, 21035234960129833544 6481) =: (e,f,f,e)$$

Take (n, u) as secrets, and publicly show others x and y.

Now we compute n and u, v from x and y.

Just compute the Jordan form of the representation matrix of x and y as in section 7.1. We get the eigenvalues of X are 0,0,46227468992939704387227,46227468992939704387227. The eigenvalues of Y are 0,0,48835863162787982086832,48835863162787982086832.

Then by computing the discrete log

$$Log(46227468992939704387227, 48835863162787982086832)$$

we get $n = 17201775578911892389118$. This is not the n at first, but we can check

$$x^{17201775578911892389118} = x^{46548060726748433200355}$$

So this is an equivalent one.

Then we may compute u and v. suppose $u = (a, b, c, d)$ now they can be calculated out by solving the linear system $ux^n = yu$. One can first compute

$$z = x^n = (17521339932500062630 7812, 24417931581393991043416,\\ 24417931581393991043416, 17521339932500062630 7812)$$

Then to find a, b, c, d we refer to the matrix equation $UZ = YU$. This is a different method from 7.1. From the form of z and y, we may first take $b = c, a = d$, then the equation above is just $48835863162787982086832b = 88708428987242022858587a$. Take $a = 1$, then a solution is found:

$u = (1, 1732491542181644967 47681, 1732491542181644967 47681, 1)$

To find v, observe that if $s = t, g = h, s + t = 1, h + g = 1$, then $r' = (s, g, h, t)$ is a right side unit of x, so v can be find by $vu = r'$. One can compute a solution using linear algebra and get $v = (113542900532618781725880, 2302577763961136457605 78, 0, 0)$.

8 conclusion

Now we have proved that HDLP and GHDLP is not as secure as it seems. It can not resist attacks from quantum computers, and even from classical computers if the order of the finite field is taken too small. In fact, we have subexponetial attack towards it.

Since we can solve HDLP and GHDLP, all the schemes based on it is never safe. We have broken a family of these schemes in the beginning of this paper, other similar schemes can also be broken using the same method with a little modification.

9 Acknowledgement

Thanks so much to my advisor Jintai Ding, who helps a lot in the structure, reference and the wording of the paper. This paper cannot be completely finished without his patient guidance.

References

[1] Dmitriy N. Moldovyan, Non-commutative finite groups as primitive of public key cryptosystems, Quasigroups and Related Systems 18 (2010), 165-176.

[2] Dmitriy N. Moldovyan, Alexandr A. Moldovyan and Nikolay A. Moldovyan, A new design of the signature schemes based on the hidden discrete logarithm problem, Quasigroups and Related Systems 29 (2021), 97-106.

[3] Alexander Moldovyan, Nicolay Moldovyan, Victor Shcherbacov, Non-commutative 6-dimensional associative algebras of two different types, Proceedings of the Conference on Mathematical Foundations of Informatics MFOI2018, July 2-6, 2018, Chisinau, Republic of Moldova

[4] A. A. Moldovyan and N. A. Moldovyan, Post-quantum signature algorithms based on the hidden discrete logarithm problem, Computer Science Journal of Moldova, vol.26, no.3 (78), 2018

[5] Dmitriy N. Moldovyan and Nikolay A. Moldovyan, Cryptoschemes over hidden conjugacy search problem and attacks using homomorphisms, Quasigroups and Related Systems 18 (2010), 177-186

[6] P. W. Shor, Polynomial time algorithms for prime factorization and discrete logarithms on quantum computer, SIAM J. Computing. 26 (1997), 1484-1509.

[7] J. Ding, D. Schmidt Rainbow, a new multivariable polynomial signature scheme, Lecture Notes Computer Sci., 3531 (2005), 164- 175.

[8] A. S. Kuzmin, V. T. Markov, A. A. Mikhalev, A. V. Mikhalev, and A. A. Nechaev. Cryptographic algorithms on groups and algebras. J. Math. Sci., 223: 629641, 2017.

[9] Vitaly Romankov, Alexander Ushakov and Vladimir Shpilrain, Algebraic and quantum attacks on two digital signature schemes. 2020 Mathematics Subject Classification. 94A60.

[10] D.Moldovyan, A. Moldovyan and N. Sklavos. Post quantum signature schemes for efficient hardware implementation. In Proceedings of the 10th IFIP International Conference on New Technologies, Mobility and Security(NTMS 2019), pages1–5. IEEE, 2019.

[11] D. N. Moldovyan, New Form of the Hidden Logarithm Problem and its Algebraic Support. Buletinul Academiei De Stiinte, A Republicii Moldova. Matematica Number 2(93), 2020, Pages 3–10, ISSN 1024–7696.

[12] N. A. Moldovyan, Finite non-commutative associative algebras for setting the hidden discrete logarithm problem and post-quantum cryptoschemes on its base, Bul. Acad. De Stiinte Republ. Moldova. Matematica, 1(89) (2019), 71-78.

[13] A. A. Moldovyan, N. A. Moldovyan, Finite non-commutative associative algebras as carriers of hidden discrete logarithm problem, Bull. South Ural State Univ., Ser. Mathematical Modelling, Programming Computer Software, 12 (2019), 66 - 81.

[14] D. N. Moldovyan, A. A. Moldovyan, and N. A. Moldovyan. Digital signature scheme with doubled verification equation, Computer Science Journal of Moldova, vol.28, no.1(82), 2020.

[15] D. N. Moldovyan, A practical digital signature scheme based on the hidden logarithm problem, Computer Science Journal of Moldova, vol.29, no. 2(86), 2021.

[16] N. A. Moldovyan, Unified Method for Defining Finite Associative Algebras of Arbitrary Even Dimensions, Quasigroups and Related Systems, vol. 26, no. 2, pp. 263-270, 2020.

[17] D. N. Moldovyan and N. A. Moldovyan, A new hard problem over non-commutative finite groups for cryptographic protocols, Lecture Notes Computer Sci., 6258(2010), 183-194.

[18] D.N. Moldovyan, A. A. Moldovyan, N. A. Moldovyan, An enhanced version of the hidden discrete logarithm problem and its algebraic support, Quasigroups and Related Systems, 28 (2020), 269-284.

[19] D. N. Moldovyan, Post-quantum public key-agreement scheme based on a new form of the hidden logarithm problem, Computer Science Journal of Moldova, vol. 27, no. 1(79), pp. 56-72, 2019.

[20] D. N. Moldovyan, N. A. Moldovyan, and A. A. Moldovyan, Commutative Encryption Method Based on Hidden Logarithm Problem, Bulletin of the South Ural State University. Ser. Mathematical Modelling, Programming Computer Software, vol. 13, no. 2, pp.54-68, 2020. DOI: 10.14529/mmp200205.

[21] V.Roman'kov. Cryptanalysis of a combinatorial public key cryptosystem. Groups, Complexity, Cryptology, 9(2):125-135, 2017.

[22] V.Roman'kov. Essays in algebra and cryptology: Algebraic cryptanalysis. Omsk State University, 2018.

[23] Tao C., Diene A., Tang S., Ding J. (2013) Simple Matrix Scheme for Encryption. In: Gaborit P. (eds) Post-Quantum Cryptography. PQCrypto 2013. Lecture Notes in Computer Science, vol 7932. Springer, Berlin, Heidelberg.

[24] Szepieniec, A., Ding, J. and Preneel, B. (2016). Extension Field Cancellation: A New Central Trapdoor for Multivariate Quadratic Systems. PQCrypto.

[25] Nguyen H.M., Moldovyan N.A., Moldovyan A.A., Nguyen N.H., Tran C.M., Phieu N.H. (2019) Post-quantum Cryptoschemes: New Finite Non-commutative Algebras for Defining Hidden Logarithm Problem. In: Cong Vinh P., Alagar V. (eds) Context-Aware Systems and Applications, and Nature of Computation and Communication. ICCASA 2018, ICTCC 2018. Lecture Notes of the Institute for Computer Sciences, Social Informatics and Telecommunications Engineering, vol 266. Springer, Cham.

[26] Moldovyan D. N., Moldovyan A. A., Moldovyan N. A. A novel method for development of post-quantum digital signature schemes. Informatsionno-upravliaiushchie sistemy [Information and Control Systems], 2020, no.6, pp. 21-29.

[27] Minh Nguyen Hieu, Moldovyan Alexander Andreevich, Moldovyan Nikolay Andreevich, and Canh Hoang Ngoc, A New Method for Designing Post-Quantum Signature Schemes. Journal of Communications Vol. 15, No. 10, October 2020.

[28] L. M. Adleman, A subexponential algorithm for discrete logarithms with applications to cryptography, Proc. 20th IEEE Found. Comp. Sci. Symp., IEEE Computer Society, Long Beach, CA, 1979, pp. 55-60.

[29] Berlekamp, E.R. (1967). Factoring Polynomials Over Finite Fields. BellSystem Technical Journal, 46(8), 1853-1859.

[30] Cantor, D. and Zassenhaus, H. (1981). A New Algorithm for Factoring Polynomials Over Finite Fields. Mathematics of Computation, 36(154), 587-592.

[31] L. Ronyai, Factoring polynomials over finite fields, 28th Annual Symposium on Foundations of Computer Science (sfcs 1987), 1987, pp. 132-137.

[32] T. Matsumoto, H. Imai, Public quadratic polynomial-tuples for efficient signature verification and message-encryption, in EUROCRYPT 1988. Lecture Notes in Computer Science, vol. 330 (Springer, Heidelberg, 1988), pp. 419–553

[33] A. Casanova, J. Faugère, G. Macario-Rat, J. Patarin, L. Perret, J. Ryckeghem, GeMSS: a great multivariate short signature. https://csrc.nist.gov/Projects/Post-Quantum-Cryptography/Round- 2-Submissions

[34] J. Patarin, The oil and vinegar signature scheme, in Presented at the Dagstuhl Workshop on Cryptography (1997)

[35] L. C. Wang, B. Y. Yang, Y. H. Hu, and F.P. Lai, Medium-field multivariate public key encryption scheme, in Proceedings of The Cryptographers' Track at the RSA Conference, pp. 132-149, Feb. 2006.

[36] J. Ding, A. Petzoldt, L.C. Wang, The cubic simple matrix encryption scheme, PQCrypto 2014, LNCS 8772 (2014), pp. 76-87.

[37] A. Petzoldt, J. Ding, L.C. Wang, Eliminating decryption failures from the simple matrix encryption scheme, http://eprint.iacr.org/2016/010, 2016.

A The Magma codes for section 7

[X := Matrix(GF(823057273411232437763981), 4, 4, [a,-b,-c,d,b,a,-d,c,c,d,a,-b,d,-c,b,a]) where a is 401061284016651457375229 where b is 140498875087816015438562 where c is 407926716616598678661430 where d is 66142058742316501906 5860;

U := Matrix(GF(823057273411232437763981), 4, 4, [a,-b,-c,d,b,a,-d,c,c,d,a,-b,d,-c,b,a]) where a is 627968683547310283890579 where b is 760824309936500718069644 where c is 523867729517403306136890 where d is 301803738536395935875170;

Y:=U*X^(4654806072674843320 0355)*U^(-1);

JordanForm(X);JordanForm(Y);]

which gives:

[54965059480992778138 9712 0 0 0]

[0 54965059480992778138 9712 0 0]

[0 0 25247197322337513336 0746 0]

[0 0 0 25247197322337513336 0746]

[61336357092527428629 4157 0 0 0]

[0 61336357092527428629 4157 0 0]

[0 0 15204106787871666371 3056 0]

[0 0 0 15204106787871666371 3056]

Then to compute using eigenvalues, the new codes are:

[F:=GF(823057273411232437763981);

a:=549650594809927781389712;

b:=613363570925274286294157;

c:=152041067878716663713056;

Log(F!a,F!b);Log(F!a,F!c);]

This gives:

[136876177176888301461603 4654806072674843320 0355]

Only the second is right because we can check the first:

[(F!25247197322337513336 0746)^136876177176888301461603;]

This is 667142049711814210926783 and do not equal to any eigenvalue of Y.

To find U, we use the null space:

[V:= Matrix(GF(823057273411232437763981), 4, 4,

[E-e,-F+f,-G+g,-H+h,F-f,E-e,H+h,-G-g,G-g,-H-h,E-e,F+f,H-h,G+g,-F-f,E-e])

where E is 794230956107611693885597

where e is 794230956107611693885597
where F is 820866556270721139098199
where f is 288671916524669674923644
where G is 163522795984598859655862
where g is 197995833127620693524665
where H is 742582741592428191070721
where h is 50359205096408517009191;
Nullspace(V);]
It gives :
[Vector space of degree 4, dimension 2 over GF(823057273411232437763981)
Echelonized basis:
(1 0 294682108297506267818653 17886283966910151881866)
(0 1 67224060003266352734558 9 11102952070305973111637 2)]