The Security of the Gabidulin Public Key Cryptosystem

Keith Gibson

Department of Computer Science, Birkbeck College, Malet Street,
London WC1E 7HX, England.
Email: jkg@uk.ac.bbk.dcs

Abstract. The Gabidulin Public Key Cryptosystem (PKC), like the well known McEliece PKC, is based on error correcting codes, and was introduced as an alternative to the McEliece system with the claim that much smaller codes could be used, resulting in a more practical system. In this paper an attack on the Gabidulin PKC is given which breaks it for codes of the size envisaged, destroying much of its advantage over the McEliece system. The attack succeeds in polynomial time for Gabidulin's choice of one of his system parameters, but it does show how to choose this parameter more appropriately. It consists of a reduction of the decryption problem for the Gabidulin PKC to consideration of a search problem that is easier to describe, and which with luck should be easier to analyse. It therefore provides a possible starting point for a proof that decryption for the Gabidulin PKC is an NP-complete problem.

1 Introduction

1.1 Algebraic Coded Public Key Cryptosystems

An Algebraic Coded Public Key Cryptosystem (PKC) is based on a family F of linear block error correcting codes with a fast decoding algorithm whose operation for each member of F requires knowledge of a key for that member. An instance I of the PKC uses a member C of F with key K. The public key for I is any generator matrix G for C, and the secret key is K. Four Algebraic Coded PKC's have been proposed:

1. McEliece[10] (1978), using Goppa codes,
2. Niederreiter[11] (1986), using generalised Reed-Solomon (GRS) codes,
3. Gabidulin[4] (1991) and [6] (1993), using Gabidulin codes,
4. Sidelnikov[12] (1994), using Reed-Muller codes.

The Neiderreiter system was broken by Sidelnikov and Shestakov[13] in 1992, though a modification suggested by Gabidulin[6] remains unbroken. The Sidelnikov system is too recent to have been evaluated. The McEliece system remains unbroken, -- the statement by Sidelnikov and Shestakov that because Goppa codes are subfield codes of GRS codes their methods can easily be adapted to break the McEliece system is just wishful thinking. The McEliece system is probably the hardest of the four systems to break precisely because it uses subfield

U. Maurer (Ed.): Advances in Cryptology - EUROCRYPT '96, LNCS 1070, pp. 212-223, 1996.

codes. An analysis of the 1991 version of the Gabidulin system has been given by the author in [7]. It is to the modified 1993 version that this paper is addressed.

1.2 The search for an NP-secure PKC

Unlike the RSA system, Algebraic Coded PKC's suffer from message expansion. The encrypted message is longer than the original, usually about twice as long, which means the sender's key cannot be kept secret, so that they cannot be used for signature schemes. They also tend to have large public keys. However the RSA system is unlikely to be NP-secure[1].

There is no known PKC for which decryption is an NP-complete problem, and it is an open question whether such a PKC exists. If one does exist it is possible that an Algebraic Coded PKC might provide an example. Although such an example would not necessarily be practically secure, since NP-complete problems are notorious for being almost always easy, there is some evidence[9] that NP-security, once obtained, can be amplified to the required cryptographic security.

1.3 Gabidulin codes and the Gabidulin PKC

Definition 1. Let x be any q-vector over $GF(2^m)$. The $p \times q$ **Gabidulin matrix** with generating vector x is the matrix whose first row is x, and whose i th row is the square of the $i-1$ th row, $i = 2 \ldots p$, powers of vectors being taken co-ordinatewise.

Let g be an m-vector over $GF(2^m)$ whose coordinates are linearly independent over $GF(2)$, and let G be the $k \times m$ Gabidulin matrix with generating vector g. The **Gabidulin code** C with generating vector g has generator matrix G, and corrects $e = (m-k)$ div 2 errors. Gabidulin in [3] and [5] gives fast decoding algorithms for C for which g acts as a key. Errors for Gabidulin codes are not counted using the usual Hamming metric, but with the rank metric induced by the rank norm $|v|$ of a vector v over $GF(2^m)$, which is defined to be the number of coordinates of v that are linearly independent over $GF(2)$. The associated rank norm $|D|$ of a matrix D over $GF(2^m)$ is the number of linearly independent columns of D over $GF(2)$. It can be shown that $|vD| \leq t$, with equality for some v, if and only if $|D| = t$.

An instance of the Gabidulin PKC using the code C is constructed by choosing a random $k \times k$ non-singular **scramble matrix** S over $GF(2^m)$, and a random $k \times m$ **distortion matrix** D over $GF(2^m)$ with rank norm $t < e$. The public key is then $Z = S(G + D)$, and the secret key is the triple (g, D, S).

The code C' with generator matrix Z corrects at least $e-t$ errors. Encryption of an information vector v is performed by encoding v with Z and adding $e-t$ random errors to obtain a received word r' of C'. It is easy to show that a decoder for C with key g, together with knowledge of D and Z, can be used to recover v from r'.

The distortion matrix D is needed because it is easy to show that if D is known then a secret key can quickly be determined. The McEliece PKC does not need a distortion matrix, but one can be used for the Niederreiter PKC, and that was Gabidulin's modification to this system referred to earlier.

The distortion matrix D can be written in the form $D = AB$, where B is a $t \times m$ binary matrix of rank t, and A is a $k \times t$ matrix over $GF(2^m)$ which has t linearly independent columns over $GF(2)$, but whose rank s over $GF(2^m)$ merely satisfies $1 \leq s \leq t$. The matrices A and B will be called the row and column distortion matrices, and the system parameters s and t the row and column distortion ranks. There may be secret keys with different values of s and t, but the minimal values of s and t are determined by Z. Secret keys with t not minimal provide only partial decoders for the code with generator matrix Z.

Both s and t play a crucial role in the analysis of the PKC. In [4] Gabidulin suggested $s = 1$, and then, when the present author [7] gave an algorithm to break medium sized instances of the PKC, suggested in [6] to take $s = t$, which turns out to be an unmitigated disaster, since the algorithm of this paper will find a secret key to such an instance in polynomial time! However the algorithm does show how to choose s more appropriately.

1.4 The trapdoor attack of this paper

There are two distinct kinds of attack on any PKC, corresponding to two distinct problems. A **trapdoor attack** solves the **trapdoor problem** by obtaining the secret key (trapdoor) from the public key, whereas a **direct attack** addresses the **decryption problem** by showing that it is computationally feasible to decrypt individual messages. This paper presents a trapdoor attack, and does not tackle the question of whether the decryption problem might be easier than the trapdoor problem.

It was to improve security against a direct attack that Gabidulin introduced his PKC. What he showed was that for comparably sized codes a direct attack would be much harder to mount for his system than for the McEliece PKC, so much smaller codes could be chosen, resulting in a more practical system with a smaller public key. Unfortunately he chose the codes so small that his PKC became vulnerable to the trapdoor attack of this paper, and a program has been written using this attack which even on a personal computer will break instances that use codes of the size he suggested very quickly. The necessary increase in the size of the code used both destroys much of the advantage claimed for the Gabidulin system, and begs the question of whether it remains practical. The attack does show however that with careful choice of system parameters the codes can be chosen small enough to produce a public key which is smaller than that of the McEliece PKC.

A very interesting side feature of the attack is that it indicates that subject to some normalisation, trapdoors to the Gabidulin PKC, at least those of the kind being sought, are in most cases uniquely determined, and this may be one of the reasons they are easier to find than expected.

The attack consists of a polynomial time reduction of the trapdoor problem to a search problem that is easier to describe, and which with luck should be easier to analyse. The reduction is in the sense that under some mild assumptions any solution to an instance of the trapdoor problem provides a solution to the corresponding instance of a special form of the search problem. It is probable that these assumptions can be removed, and also probable that the special search problem is equivalent to a sub-problem of the trapdoor problem. "All" that is required therefore is to show that the special search problem is NP-complete and almost always hard, and then to show that decryption is at least as hard as finding a secret key. If that could be done it would certainly be worth finding a way of coping with any impracticalities of the Gabidulin PKC.

2 Summary of Results

2.1 On breaking the Gabidulin PKC

Consider an instance of the Gabidulin PKC of length m, dimension k, generating vector g with corresponding generator matrix G, row and column distortion matrices A and B, minimal row and column distortion ranks s and t, scramble matrix S, and public key $Z = S(G + AB)$. The breaking algorithm takes Z as input and returns the secret key (G, A, B, S) as output. In the ensuing discussion t is assumed given, but this is not in fact necessary. Without loss of generality B may be sought in row echelon form.

Write $n = k + t + 2$. It is necessary to assume the conditions $n \leq m$ and $t + 2 \leq k$, but note that if $n > m$ then the code P with generator matrix Z corrects no errors, and if $t + 2 > k$ then either P corrects no errors or $m \geq 3k$.

The cost of the algorithm depends on a quantity called the **deficiency** d of Z. In all cases $0 \leq d \leq t$, and computational evidence strongly suggests that $d \geq max(0, t - 2s)$, with equality almost always. It is defined with respect to a systematic form of a selection of n columns of Z, and does not appear to depend on which columns are selected. Suppose therefore that Z is in systematic form to start with.

Define the **partial column distortion matrix** C to be the first n columns of B. The bulk of the work of the breaking algorithm is to find C, and the main theorem of this paper is the one that shows how to reduce finding C to a search problem. It will be assumed throughout that C has rank t. Discussion is made more complicated when C does not have rank t, but it is then actually easier to find a trapdoor. The following notation is used:

1. For any integer k, I_k denotes a $k \times k$ identity matrix.
2. For any matrix X over $GF(2^m)$, $X^{(2)}$ denotes the result of squaring each element of X.
3. For any matrix X over $GF(2^m)$, $N(X)$ denotes a matrix whose columns form a basis of the null space of X, i.e., the set of column vectors v with $Xv = 0$.

Definition 2. Let $[I_k, U]$ be the first n columns of Z.
Write $V = U + U^{(2)}$, and let V have rank r.
The **deficiency** of Z is $d = t + 1 - r$.

Theorem 3 The Main Theorem.

Write $W = \begin{bmatrix} U \\ I_{t+2} \end{bmatrix} N(V)$, *so that* W *is an* $n \times (d+1)$ *matrix of rank* $d+1$.

Then under some mild assumptions, CW *has rank* d.

The search problem with parameters $(n, t, d), 0 \leq d \leq t \leq n/2 - 2$.

1. The general form.
   Given an $n \times (d+1)$ matrix X over $GF(2^m)$ of rank $d+1$.
   Find a $t \times n$ binary matrix D of rank t such that the rank of DX is d.
2. The special form.
   Given an $(n-t-2) \times (t+2)$ matrix X over $GF(2^m)$ such that $Y = X + X^{(2)}$ has rank $t + 1 - d$.
   Find a $t \times n$ binary matrix D of rank t such that $D \begin{bmatrix} X \\ I_{t+2} \end{bmatrix} N(Y)$ has rank d.

Brassard[1] shows that a proof that a problem is both NP-complete and in Co-NP is a proof that NP = Co-NP. However the search problem seems unlikely to be in Co-NP. A given instance of either form of the problem does not necessarily have a solution, and there seems no easy way of determining when it does not. The corresponding decision problem can just be: given an instance of the search problem, does it have a solution?

Theorem 4 The Breaking Theorem.

1. *Finding* C *can in most cases be reduced with* $O(k^3)$ *multiplications to an instance of the special search problem with parameters* (n, t, d).
   The reduction is in the sense that C *provides a solution to this instance.*
2. *This instance can be solved at a cost of* $O(nd2^{d(k+2)})$ *multiplications. For* $d = 1$ *and* $d = 2$, *the cost can be improved to* $O(n^d 2^{d(k+2)})$ *additions with storage for* $O(n^{d+1})$ *m-bit integers. For* $d = 0$ *the cost is* $O(n^2)$ *additions.*
3. *Once* C *is known, the remainder of the secret key can be found with* $O(k^3 + (m-k)t2^d)$ *multiplications.*

There is strong computational evidence that the row echelon form of B is uniquely determined, and indeed that when $d = max(0, t - 2s)$ the search problem has a unique solution in row echelon form. When it did not there were relatively few solutions, and just one of them enabled a full secret key to be found. All of them provided a partial secret key for the first n columns of Z, and since these columns are the input to the special search problem the indication is that the special search problem is equivalent to the trapdoor problem restricted to codes of length n.

In [6], Gabidulin took $s = t$, when for almost all instances the deficiency d will be 0, and the cost just $O(k^3)$ multiplications! He suggests $m = 20, k = 10, t = 3$, and $s = 3$. It takes just 2 seconds to break such an instance on a personal computer. When $d = 1$, the algorithm is still very fast. If in the example above s is chosen to be 1 as in the 1991 version of the PKC, then d will in most cases be 1, and the instance will still be broken in about 2 seconds.

2.2 On the minimum size of code needed for the Gabidulin PKC

Consider again an instance of the Gabidulin PKC with the parameters described in Section 2.1. It corrects $e = (m - k)$ div $2 - t$ errors, and the examples in [6] have $e = 2$, which seems a bit small to guard against a direct attack. In the example given below, the heuristic $2^e > k$ has been adopted, since this means that if a systematic generator matrix is chosen as public key information symbols in a codeword can then be hidden by distinct noise coordinates when performing encryption.

Theorem 5. *An instance with $m = 48, k = 24, s = 2$, and $t = 7$ will take about 2^{85} multiplications over $GF(2^{48})$ to break using the breaking algorithm for the modified PKC, and about 2^{112} multiplications using an extended version of the breaking algorithm for the original PKC given in [6].*

This example will have a public key of 56,000 bits, half that if given in systematic form, which compares with 500,000 bits for an instance of the McEliece PKC with the parameters suggested in [10]. It will have a deficiency of $d = 3$, and correct $e = 5$ errors. It should be regarded as having the absolute minimum size acceptable, and it would probably be better to choose $m = 64, k = 32, s = 2$, and $t = 7$, giving $d = 3, e = 9$, and a public key of 131,000 bits.

3 Some Technical Lemmas

At the heart of the main theorem is a technical lemma called the Matrix Update Lemma, whose proof seems to be difficult, and which has some interest in its own right. Proofs of the other two lemmas of this section are relatively easy and are omitted. The notations $X^{(2)}$ and $N(X)$ are those of Section 2.

Lemma 6. ***The Matrix Update Lemma***
Let K be a non singular $k \times k$ matrix over $GF(2^m)$, $J = K^{-1}$, and $L = J + J^{(2)}$.
Let B be a binary $k \times k$ matrix, and suppose $K^* = K + B$ is invertible.
Define corresponding J^* and L^* in the obvious way.
Suppose L and L^* have the same rank r, and that there is a permutation π of the columns of L and L^* for which the first r columns of $L\pi$ and $L^*\pi$ are independent.
Then there is an invertible matrix R such that $J^*N(L^*)R = JN(L)$, from which it follows that $N(L^*)R = (I_k + BJ)N(L)$.

Method of proof

The Sherman Morrison matrix update formula [2] provides an explicit formula for J^* in terms of J in the case when B has just one non-zero element, and can be used to prove the lemma for this case. To lift the result to the general case requires the assumption that K can be transformed to K^* with a succession of additions of 1 to one element while preserving the invertibility of K and the conditions on L and L^* at each stage. This is a long and unsatisfactory proof involving an additional assumption which is almost certainly unnecessary. Details can be found in [8].

Lemma 7. ***The rank 1 lemma***
Let G be a $k \times k$ Gabidulin matrix with generating vector g, let H be a $k \times f$ Gabidulin matrix with generating vector h, and suppose the vector $[g, h]$ has independent coordinates over $GF(2)$, so that G is non-singular. Let $X = G^{-1}H$. Then $X + X^{(2)}$ has rank 1, and has no zero elements.

Lemma 8. ***The null space lemma***
Let A and B be matrices over any field, each with the same number of columns. Then $rank(AN(B)) - rank(BN(A)) = rank(A) - rank(B)$.

4 The Breaking Algorithm

4.1 Overview

The object of this section is to prove the main theorem of Section 2, and to indicate how it is used to obtain the breaking theorem. There is a final paragraph on the solution of the search problem, but an efficient solution must form the subject of a separate paper. Details have been given in[8].

For the whole of this section the convention is adopted that matrix blocks are suffixed with the number of columns that they have. As in Section 2, $N(X)$ is a matrix whose columns form a basis of the null space of the matrix X, and $X^{(2)}$ is obtained by squaring each element of X.

Consider an instance of the Gabidulin PKC as described in Section 2. The breaking algorithm proceeds in three stages, but the story is told backwards, because that is the way the algorithm unfolds. In stage 3, the column distortion matrix B is assumed known, and the rest of the secret key is recovered. The method of doing so sets the framework on which stages 2 and 1 build. In stage 2, only the partial column distortion matrix C is known, and the method of stage 3 is extended using the rank 1 and null space lemmas to recover the rest of the column distortion matrix. Finally, in stage 1 only the public key is known, and the method of stage 2 is further extended using the matrix update lemma to dig out the partial column distortion matrix.

A number of assumptions have to be made, all of which are probably either provable or unnecessary. The first is probably unnecessary, but simplifies the discussion, and does in fact hold most of the time anyway.

Assumption 1 *Any selection of n columns of z are independent.*

4.2 Stage 3: The column distortion matrix is known

Stage 3 starts with the observation that since B is binary, the $m \times (m-t)$ matrix $N(B)$ can be taken to be binary, so that if $H = GN(B)$ then H is a $k \times (m-t)$ Gabidulin matrix whose generating vector h has linearly independent coordinates over $GF(2)$, and $ZN(B) = SH$. Summarising the relations between the matrices defined so far,

$$Z = S(G + AB), \quad H = GN(B), \quad ZN(B) = SH. \tag{1}$$

Assume from now on that Z is in systematic form, and that columns $k+1 \ldots k+t$ of B form an identity matrix. Since C has rank t, assumption 1 guarantees that this normalisation is possible. Write $f = m - k - t$, and partition Z, G, g, H, h, and B as follows :

$$\begin{array}{ll} Z = [I_k, Z_t, Z_f], & B = [B_k, I_t, B_f], \\ G = [G_k, G_t, G_f], & g = [g_k, g_t, g_f], \\ H = [H_k, H_f], & h = [h_k, h_f]. \end{array} \tag{2}$$

Take

$$N(B) = \begin{bmatrix} I_k, & 0 \\ B_k, & B_f \\ 0, & I_f \end{bmatrix} \begin{matrix} k \\ t \\ f. \end{matrix} \tag{3}$$

With this partitioning, (1) yields

$$H_f = H_k X, \tag{4}$$

$$[g_k, g_f] = h + g_t\,[B_k, B_f], \tag{5}$$

$$A = G_t + H_k U_t, \tag{6}$$

$$Q = SH_k, \tag{7}$$

where

$$\begin{array}{l} Q = I_k + Z_t B_k, \\ P = Q^{-1}, \\ U = P[Z_t, Z_f] = [U_t, U_f], \\ X = U_f + U_t B_f. \end{array} \tag{8}$$

To solve (4) for h observe that g, and hence h, is only determined up to a scalar multiple, so assume that h has been normalised so that the first coordinate of h_f is 1. Let x_1 be the first column of X, and X_1 the $k \times k$ Gabidulin matrix whose generating vector is the 2^{m-k-1} th power of x_1, taken coordinatewise. Let e denote a k-vector of ones. Then after some manipulation, (4) implies $X_1 h_k = e$, which determines h_k and hence h, provided X_1 is non singular. The condition that X_1 is non singular is that the coordinates of x_1 are independent over $GF(2)$, and in view of (9) and (11) below, this requires

Assumption 2 *the rows of X are independent over $GF(2)$.*

Once h is known, (5) can be used to determine g, (6) to determine A, and (7) to determine S. In (5), g_t can be chosen randomly subject to ensuring that g has independent coordinates over $GF(2)$, and this flexibility of choice can be used to ensure A has its minimal rank s, which determines g uniquely when $d = max(0, t - 2s)$. Note that (7) ensures that Q is invertible, since H_k and S are invertible.

4.3 Stage 2 : The partial column distortion matrix is known

In terms of the previous section, B_k and the first two columns of B_f are known, and the remaining columns of B_f have to be determined. From the definition of X in (8),

$$X^{(2)} + X = VY, \tag{9}$$

where

$$V = U + U^{(2)}, \quad Y = \begin{bmatrix} B_f \\ I_f \end{bmatrix} \begin{matrix} t \\ f \end{matrix}. \tag{10}$$

Equation (4) states that $H_f = H_k X$, so applying the rank 1 lemma gives

Lemma 9. ***The Stage 2 lemma***

V Y has rank 1, and has no zero elements. (11)

Suppose V has rank r, and let $d = t + 1 - r$. The columns of Y form a basis of the null space of $[I_t, B_f]$, so applying the null space lemma gives

Theorem 10 The Stage 2 Theorem.

$[I_t, B_f]N(V)$ *is a* $t \times (d + f - 1)$ *matrix of rank* d. (12)

This theorem is used to determine B_f. The method requires

Assumption 3 *The first t columns of V span the column space of V, and no selection of d columns of $[I_t, B_f]N(V)$ are dependent.*

Assumption 3 means that the last $d + f - 1$ rows of $N(V)$ can be taken to be an identity matrix. Let b be the first column of B_f, and c be any other column. Then for a suitable known $r \times (d + 1)$ matrix W, equation (12) gives

$$[I_t, b, c] \begin{bmatrix} W \\ I_{d+1} \end{bmatrix} \begin{matrix} r \\ d+1 \end{matrix} \text{ has rank } d. \tag{13}$$

The vector b is known, and for $d < 2$ equation (13) determines c very quickly. For $d \geq 2$ it determines c after a search over the last $d - 1$ coordinates of c. Thus when the first column of B_f is known, the Stage 2 theorem can be used to find each of the remaining columns of B_f in turn, at a total cost of $O(t(m - k)2^d)$ multiplications. Further details are given in [8].

It can be shown that due to the special form of Y the Stage 2 lemma implies

Lemma 11. ***The Stage 2 Corollary***

The first $t+1$ columns of V do span the column space of V.

This corollary will be used to show that under assumptions made in stage 3, the quantity d is the deficiency of Z as defined in Section 2. Note that the Stage 2 theorem implies that $1 \leq r \leq t+1$, so that $0 \leq d \leq t$.

4.4 Stage 1 : Only the public key is known

Stage 1 finds the first $n = k+t+2$ columns of B, and therefore restricts attention to the first n columns Z, so that the f of stages 2 and 3 is $f = 2$, but the symbol f will continue to be used to indicate the number of columns in matrix blocks. Write $p = t+2$ and $q = k-p$. The conditions $n \leq m$ and $t+2 \leq k$ ensure that there are n columns of Z to play with, and that $q \geq 0$.

In stage 1, the matrix V is not available since B is not known. The basic idea of stage 1 is to let $V0$ be the value of V obtained by setting $B = 0$, and to use the matrix update lemma to see how V and $V0$ are related. First some useful matrix blocks are defined. Note that since Z is in systematic form the block Z_q consists of q columns of Z. Define

$$D_k = \begin{bmatrix} B_k \\ 0 \end{bmatrix} \begin{matrix} t \\ k-t \end{matrix}, \quad E_k = \begin{bmatrix} B_k \\ 0 \end{bmatrix} \begin{matrix} t \\ 2 \end{matrix}, \quad Z_q = \begin{bmatrix} 0 \\ I_q \end{bmatrix} \begin{matrix} p \\ q \end{matrix}. \tag{14}$$

Write

$$E_k = [E_p, E_q]. \tag{15}$$

The first step is to define $V0$, and build it up to a square matrix so that the matrix update lemma can be applied. With the matrix P of equation (8), define

$$\begin{array}{ll} U0 = [Z_t, Z_f], & U = P\ U0, \\ V0 = U0 + U0^{(2)}, & V = U + U^{(2)}, \\ J0 = [U0, Z_q], & J = P\ J0, \\ K0 = J0^{-1}, & K = J^{-1}, \\ L0 = J0 + J0^{(2)}, & L = J + J^{(2)}. \end{array} \tag{16}$$

The matrices U and V defined here consist of the first $t+2$ columns of the U and V defined in (8) and (10). The matrices $U0$ and $V0$ are the U and V of section 2. Note that $J0$ is invertible by assumption 1, since it consists of k columns of Z. After some manipulation, (8) and (16) yield

$$K = K0 + D_k, \tag{17}$$

$$\begin{array}{ll} \overset{p}{L0} = [\overset{p}{V0}, \overset{q}{0}]k, & \overset{p}{L} = [\overset{p}{V}, \overset{q}{V\ E_q}]\ k. \end{array} \tag{18}$$

The stage is now set for the matrix update lemma, but an assumption must be made to ensure it can be applied.

Assumption 4 *V and $V0$ have the same rank r, and there is a permutation π of the columns of V and $V0$ such the first r columns of V π and $V0$ π are independent.*

Take

$$N(L0) = \begin{matrix} p-r & q \\ \left[\begin{matrix} N(V0), & 0 \\ 0, & I_q \end{matrix}\right] & \begin{matrix} p, \\ q \end{matrix} \end{matrix} \qquad N(L) = \begin{matrix} p-r & q \\ \left[\begin{matrix} N(V), & E_q \\ 0, & I_q \end{matrix}\right] & \begin{matrix} p \\ q \end{matrix} \end{matrix} \tag{19}$$

The matrix update lemma now says there is an invertible $k \times k$ matrix R with $N(L)R = (I_k + D_k\ J0)N(L0)$, and sorting out the matrix blocks in this expression shows that there is an invertible $p \times p$ matrix T with

$$N(V)T = (I_p + E_k\ U0)N(V0). \tag{20}$$

Now it is straightforward to see that

$$[I_t, B_f](I_p + E_k\ U0) = [B_k, I_t, B_f] \begin{matrix} \left[\begin{matrix} U0 \\ I_p \end{matrix}\right] & \begin{matrix} k \\ p. \end{matrix} \end{matrix} \tag{21}$$

The final step is to use the Stage 2 theorem, but this theorem was developed using all m columns of Z. However it remains true when only n columns are used, and states, with the Stage 1 definition of V and r, that $[I_t, B_f]N(V)$ has rank $d = t + 1 - r$. Further, the Stage 2 corollary together with assumption 4 ensures that the quantities d and r defined in Stages 1 and 2 are the same, and that d is the deficiency of Z as defined in section 2. The Stage 2 theorem, together with (20) and (21), gives

Theorem 12 The Stage 1 Theorem.

$$C \begin{bmatrix} U0 \\ I_p \end{bmatrix} N(V0) \text{ has rank } d = t + 1 - r. \tag{22}$$

A closer examination reveals that provided C does have rank t this result remains true even when columns $k+1 \ldots k+t$ of C do not form an identity matrix, at least if assumption 1 holds, and with this observation the main theorem of Section 2 is proved.

4.5 Solving the general search problem

Given an $n \times (d+1)$ matrix X over $GF(2^m)$ of rank $d+1$, it is required to find a $t \times n$ binary matrix D of rank t such that the rank of DX is d, $0 \leq d \leq t \leq n/2-2$. The method of searching for D given in [8] requires an assumption which can be translated as one final assumption about the matrices C and $U0$ in (22).

Assumption 5 *The first d rows and columns of DX form an invertible matrix.*

D can be sought in row echelon form, and with assumption 5 it is enough to search over the first d rows and $n-t+d$ columns of D. Thus the search is over $d \times (n-t+d)$ binary row echelon matrices of rank d, and there are fewer than $3.5 * 2^{d(n-t)}$ of these. The work can be cut to $O(nd)$ multiplications per matrix tested by enumerating the matrices so that each differs from the previous one in just one element, and using the Sherman Morrison formula [2] as in the proof of the matrix update lemma. To finish on an aesthetically pleasing note, this enumeration is an adaptation of the following way of enumerating r-bit integers so that each differs from the previous one in just one bit position. Let $s = 2^r - 1$ and number the integers $a_0 \ldots a_s$, with any integer assigned to a_0. For $i = 0$ to $s-1$ let j be minimal with bit j of i equal to zero, and obtain a_{i+1} by complementing bit j of a_i.

References

1. BRASSARD, G. "A Note on the Complexity of Cryptography." IEEE Transactions on Information Theory, Vol IT-25, no. 2, 1979.
2. BURDEN R.L., FAIRES J.D., and REYNOLDS A.C. "Numerical Analysis." 2nd. Ed., Prindle, Weber, and Schmidt, 1981. Page 458.
3. GABIDULIN E.M. "Theory of Codes with Maximum Rank Distance." Problems of Information Transmission, Vol 21 no. 1, 1985.
4. GABIDULIN E.M. "Ideals Over a Non-Commutative Ring and their Applications in Cryptography." Lecture Notes in Computer Science Vol 547, Proc. Eurocrypt 91, Springer Verlag, 1991.
5. GABIDULIN E.M. "A Fast Matrix Decoding Algorithm for Rank-Error-Correcting Codes." Lecture Notes in Computer Science Vol 573, Algebraic Coding, Springer Verlag, 1992.
6. GABIDULIN E.M. "On Public-Key Cryptosystems Based on Linear Codes : Efficiency and Weakness." Codes and Ciphers, Proc. 4th IMA Conference on Cryptography and Coding, 1993. IMA Press, 1995.
7. GIBSON J.K. "Severely Denting the Gabidulin Version of the McEliece Public Key Cryptosystem." Designs, Codes, and Cryptography, Vol 6, 1995.
8. GIBSON J.K. "Algebraic Coded Cryptosystems". PhD Thesis, Univ. of London, 1996.
9. GOLDREICH O., IMPAGLIAZZO R., LEVIN L., VENKATESAN R., and ZUCKERMAN D. "Security Preserving Amplification of Hardness." Proc. of the 31st Annual Symposium on the Foundations of Computer Science (FOCS), 1990.
10. McELIECE R.J. "A Public Key Cryptosystem Based on Algebraic Coding Theory". DSN Progress Report (Jan-Feb), Jet Propulsion Laboratory, California Institute of Technology, 1978.
11. NIEDERREITER H. "Knapsack-Type Cryptosystems and Algebraic Coding Theory." Problems of Control and Information Theory, Vol 15 no. 2, 1986.
12. SIDELNIKOV V.M. "A Public-Key Cryptosystem Based on Binary Reed-Muller Codes." Discrete Mathematics and Applications, Vol 4, no. 3, 1994.
13. SIDELNIKOV V.M. and SHESTAKOV S.O. "On Insecurity of Cryptosystems Based on Generalised Reed-Solomon Codes." Discrete Mathematics and Applications, Vol 2, no. 4, 1992.