Matrix Methods for Securely Sending Messages and Establishing Cryptographic Keys

FRANK RUBIN *

Abstract — **New methods are introduced for two parties to exchange messages and to establish cryptographic keys without the need to distribute secret keys beforehand. The methods rely on establishing a commutative family F of invertible square matrices over a non-commutative ring. To exchange messages, the sender and receiver independently choose encryption matrices S and R from F. The message vector is successively multiplied by S, R, S' and R'. Due to the commutativity property, this recovers the original message. For establishing cryptographic keys, a random vector V is chosen. The sender transmits SV to the receiver, and the receiver transmits RV to the sender. This lets both parties compute SRV=RSV, which is then used as the cryptographic key.**

The new matrix algorithm is about 2000 times as fast as the current exponentiation method. Likewise, it is thousands of times as fast as public key encryption based on exponentiation.

Index Terms— **commutative family, cryptography, decryption, encryption, key distribution, matrix, non-commutative ring, security, three-pass protocol**

I. INTRODUCTION

There is a story of a king who wished to send a valuable gift to a neighboring princess. He had an impregnable strongbox and a pickproof lock, but he could not send the key with the messenger, nor even with a second messenger for fear the two could join up along the route, open the box and steal the gift. The solution was for the princess to add her own unpickable lock to the strongbox, and send it back. Then the king removed his lock, and sent the box back with only her lock.

* * Manuscript received Nov. 1, 2007.The author is with Master Software Corporation, Wappingers Falls, NY 12590

This is the basic principle of *Private Key Cryptography*. Each party has a private encryption key and its inverse decryption key, which are not known by or shared with any other party. This contrasts with traditional *Secret Key Cryptography* where both correspondents share a secret key, and more recent *Public Key Cryptography* where each party has both a private decryption key and a public encryption key known to everyone.

The great advantage of Private Key Cryptography is that no infrastructure is required. It needs no key distribution network, no trusted key authority, no key-encrypting master key. Two parties who have made no prior arrangements, and who share no secret information, may establish secure communications over public channels using openly available hardware or software.

A. Sending messages

Messages are sent using a *3-Pass Protocol* [1,3]. It consists of the sender encrypting the message M with the sender's private encryption key S, and transmitting this encrypted message SM to the receiver. The receiver super-encrypts SM with the receiver's private encryption key R, and sends this doubly-encrypted message RSM back to the sender. The encryption functions are chosen so that they commute. That is, RSM=SRM. This allows the sender to decrypt RSM with the sender's private decryption key S', leaving S'RSM=RM encrypted with only the receiver's key. This partly-decrypted message is sent back to the receiver. The receiver decrypts the message with the receiver's private decryption key R', thus obtaining the original unencrypted message R'RM=M.

B. Establishing cryptographic keys

A similar method can be used to establish cryptographic keys. The basic idea is for the sender to take a random message M and to encrypt it with the sender's private encryption key S. The message M and encrypted message SM are sent to the receiver. The receiver encrypts M with the receiver's private key R and sends RM back to the sender. Again, the encryption functions S and R are chosen so that they commute. This allows both the sender and receiver to obtain SRM=RSM which is used as the cryptographic key.

The encryption and decryption functions currently used are based on exponentiation. This paper introduces new Private Key Encryption

methods using the 3-Pass Protocol with commutative matrices over a non-commutative ring. The new matrix methods are typically 2000 times as fast as the old exponentiation method.

To show that this speed really can be achieved in practice, the following chapters will select a sample ring, determine how large the matrices must be to obtain the desired level of security, generate the matrices, and finally run a timing comparison against a commercial version of the exponentiation method.

II. Exponentiation Method

The first commutative encryption [1] used exponentiation. The message is treated as an integer M modulo a large prime p. Encryption and decryption both consist of raising the integer M to a large power modulo p. Let s and s' be the sender's encryption and decryption exponents, and let r and r' be the receiver's encryption and decryption exponents, with $ss' \equiv rr' \equiv 1 \pmod{p-1}$. By Fermat's theorem [2] $M^{ss'} \equiv M^{rr'} \equiv M \pmod{p}$. The sender transmits $M^s \pmod{p}$ to the receiver, the receiver transmits $(M^s)^r \equiv M^{sr} \pmod{p}$ back to the sender, the sender transmits $(M^{sr})^{s'} \equiv (M^{ss'})^r \equiv M^r \pmod{p}$ to the receiver, who decrypts it as $M^{rr'} \equiv M \pmod{p}$. This method is little-used in practice since raising a number to a large power modulo a large prime is such a slow operation.

The Massey-Omura method [3] achieves faster speed by using multiplication in a Galois Field $GF(2^m)$ instead of multiplying integers modulo a large prime. This improves the speed, but the method still requires raising large numbers to large powers, so it is still slow.

The same methods were used for establishing keys [8,9]. A large prime p and a primitive root w of p are chosen beforehand. To establish an encryption key the sender and receiver randomly choose exponents s and r in the range 2 to p-2. The sender sends $w^s \pmod{p}$ to the receiver, and the receiver sends $w^r \pmod{p}$ to the sender. Then the common value $w^{rs} \equiv w^{sr} \pmod{p}$ is used as the cryptographic key.

The exponentiation method is believed to be as difficult to solve as the Discrete Logarithm Problem [7].

III. Matrix Methods

A message is a string of characters over a finite alphabet. For simplicity, we will assume that there is a one-to-one correspondence between the symbols of the alphabet and the elements of some non-commutative ring $\boldsymbol{R}$. That is, the alphabet and the ring were chosen to be the same size. If not, then there would need to be extra steps to convert the message characters to ring elements, and back. The message will be encrypted and decrypted in blocks of bc characters each, with each block treated as a b×c matrix over the ring $\boldsymbol{R}$. Let F be a large commutative family of invertible b×b matrices over the ring $\boldsymbol{R}$. Such commutative families exist for every ring. The best-known example is the family of diagonal matrices. The following section will describe methods for producing other families. It is expected that the choices for $\boldsymbol{R}$ and F would be built into the software package or hardware device.

For each block M of the message the sender randomly chooses an encryption matrix S from F and the receiver randomly chooses an encryption matrix R from F. Left-side encryption matrices will be used in this paper, but right-side matrices are equally valid and equivalent. Let the inverse matrices be S' and R'. The sender will encrypt the message block M as SM and send this to the receiver The receiver will super-encrypt the block as RSM and send this back to the sender. The sender will then remove the S encryption by S'RSM = S'SRM = RM and send this back to the receiver. The receiver can then remove the remaining encryption by R'RM = M.

For establishing keys, the sender chooses a random message block M, usually a vector, and a matrix S from F and sends both M and SM to the receiver. The receiver then computes RM and RSM and sends RM back to the sender. The sender can now compute SRM = RSM which will be used as the cryptographic key.

This establishes that the matrix methods work. What still must be shown is that the methods are practical and secure. They will be practical only if it is feasible to find large commutative families of matrices.

A. Commutative families of matrices

The best-known commutative family of square matrices is the family of diagonal matrices. If the diagonal elements are commutative, then the matrices will commute. If every element on the diagonal is invertible, then the matrix will be invertible. However, using diagonal matrices clearly will not lead to a secure encryption.

Another well-known way to construct a commutative family of matrices is to choose an arbitrary invertible base matrix B and take its successive powers B, B^2, B^3, … Since matrix

multiplication is associative, this family of matrices is commutative. Eventually B^k will equal I, the identity matrix, where k is the order of the matrix in the multiplicative group of matrices, so k is the size of the family. A suitable choice of the base matrix can produce a very large commutative family. (To avoid confusion, the *order* of a matrix will always refer to its multiplicative order; the number of rows and columns will be called the *size* of the matrix.)

A third way to construct a commutative family of matrices is to start with any invertible matrix A and solve the commutativity equation AM=MA. When a solution is found, say B, then the simultaneous matrix equations AM=MA and BM=MB can be solved. Any solution will commute with both A and B. After just a few steps, the reduced row echelon form of the matrix equation will stabilize. Adding additional commutativity conditions results in an equivalent matrix. This final set of linear equations will generate a locally maximal commutative family of matrices by using back substitution.

Once one commutative family F has been found, additional families can easily be constructed. If X is any invertible matrix, and A and B commute, then XAX' and XBX' commute. So XFX' = {XAX' : A∈ F} is also a commutative family of the same cardinality.

IV. Security

Let us now examine the new encryption method from the standpoint of an opponent who wishes to read an intercepted message. It is safest to assume that the opponent has complete knowledge of the system, including knowing the ring ***R*** and the commutative family F, and that the opponent has intercepted all 3 transmissions, SM, RSM and RM. Call these intercepted messages X, Y and Z. The opponent can read the message by determining either S or R. The easiest way to do that is from the relationships Y=RX and Z=S'Y. For the key establishment protocol the known values of M, SM and RM are similarly used.

A. Commutative ring

To motivate the discussion, consider what the opponent would do if the ring were commutative. The matrix R contains b^2 elements which are unknown to the opponent. Since the opponent has intercepted X and Y, the relationship RX=Y provides bc equations in these b^2 unknowns. When c<b, this is not enough information to solve for the b^2 unknowns. However, the opponent also knows that R is in the family F. If the opponent chooses any matrix A from the family F, then AR=RA. This provides b^2 linear equations in the b^2 unknowns in R. This $b^2 \times b^2$ matrix will be called the *commutativity matrix* of F.

These b^2 equations are not linearly independent. When the commutativity matrix is reduced by the standard method of Gaussian elimination, it will yield b(b-d) linearly independent equations, where d depends upon the choice of the ring ***R*** and the family F.

When F is a maximal family, that is, F has maximum cardinality among all commutative families for ***R***, and when the opponent has made a good choice for the matrix A, then d will be 1, and it will be easy for the opponent to solve for the matrix R. The b(b-1) equations from RA=AR and the bc equations from RX=Y' are enough to solve for the b^2 unknowns even when c is as small as 1. This is sufficient for the opponent to recover M.

The sender has two defenses against such an attack. The first is to reduce the rank of the equation RA=AR, that is to increase the value of d. The second is to decrease the rank of RX=Y.

When ***R*** is the ring of integers modulo a prime p, the maximum multiplicative order for an n×n matrix over ***R*** is p^n-1. Every maximal commutative family F of b×b matrices over ***R*** will have cardinality $k=p^b-1$, and can be generated as the powers of some base matrix B, namely B, B^2, B^3, …, B^k=I.

If b is composite, say b=fd, then the polynomial x^b-1 can be factored as $(x^f-1)(x^{n-f}+x^{n-2f}+\ldots+1)$. If the base matrix B is replaced by the base matrix B^h where $h=(x^{n-f}+x^{n-2f}+\ldots+1)$, then the commutative family F^* of matrices will have cardinality p^f-1. The commutativity matrix of F^* will have rank b(b-d).

It would seem like the sender could defeat the opponent by choosing F* as above, and using a message matrix of row rank r such that $b(b-d)+rc+c(c-e)<b^2+c^2$, namely r<(bd+ce)/c. Then the rank of the message matrix will be too low to permit the opponent to solve for the matrices R.

Unfortunately, this does not work. It is not necessary for the opponent to recover the original matrix R in order to read the message. Any matrix Q satisfying QA=AQ and QX=Y will allow the opponent to recover the message M.

The encryption is not secure using matrices over a commutative ring.

B. Non-commutative ring

Next consider the situation when the ring $\boldsymbol{R}$ is not commutative. The opponent begins the attack on the message as before. The equations AR=RA and RX=Y still provide linear equations. Since linear equations are the easiest to solve, this is again the best starting point for the attack. The equations will now be of the form $\sum_j A_{ij}R_{jk}=\sum_j R_{ij}A_{jk}$. The A_{ij} are known constants from the chosen matrix A, while the R_{ij} are the unknown elements from the receiver's encryption matrix R. Similarly for X and Y. Such equations are sometimes called *bilinear*. Since ring multiplication is not commutative, the terms on the left side and the terms on the right side of each equation usually cannot be combined. At first, the problem looks intractable.

The best approach is to convert the bilinear equations to linear equations. This is called *linearization*. The basic idea is to replace the unknowns R_{ij} in the equations with an expanded set of unknowns [4] such that every term in the expanded set of equations can be written as sX, where s is a scalar element of $\boldsymbol{R}$ and X is one of the expanded set of unknowns. In order to make this work, a new representation must be found for the elements of $\boldsymbol{R}$. Two such representations will be considered.

In the first representation, each element of $\boldsymbol{R}$ will be represented as a product su_i where s is a commutative element of $\boldsymbol{R}$ and $U=\{u_1,u_2,\ldots,u_g\}$ is a set of generators for the elements of $\boldsymbol{R}$, with $u_1=1$ and the other generators non-commutative. In general, different sets of generators will have different cardinalities. The representation of each scalar as su_i is not unique. It seems to be easy to find minimal generator sets. Once U has been chosen, a term $R_{ij}A_{jk}$ can be replaced by $R_{ij}su_k$ for some s and u_k. Since s is commutative, $R_{ij}su_k=sR_{ij}u_k$. This puts the scalar coefficient s to the left of the term $R_{ij}u_k$. The set of b^2g terms $R_{ij}u_k$ will be the expanded set of unknowns. The equations are now in linear form with this set of variables.

The second representation expresses each scalar in $\boldsymbol{R}$ as a sum $a_1v_1+a_2v_2+\ldots+a_hv_h$ where the a_i are commutative elements of $\boldsymbol{R}$ and $V=\{v_1,v_2,\ldots,v_h\}$ is a set of generators, with $v_1=1$. This representation is analogous to representing complex numbers in the form a+bi where a and b are real numbers and i is a generator. As before, each of the commutative coefficients a_i can be moved to the left in each term, making the equations linear in the expanded set of unknowns $R_{ij}v_k$ and $Q'_{ij}v_k$.

The advantage of the first representation is that each scalar is represented as a single term rather than a sum of terms. This greatly simplifies the multiplication of scalars. In the second representation a product contains h^2 terms which must be combined. However, h will normally be smaller than g, so that the second representation involves fewer equations in fewer unknowns. Since the work involved in solving the set of equations is proportional to the number of equations times the square of the number of unknowns, the opponent may be expected to choose V instead of U.

The change of representation gives b^2+bc equations in b^2h unknowns. This is not enough to solve them. The trick is to augment the set of equations by right-multiplying each equation by $v_2, v_3, \ldots, v_h$ in turn. The original equations plus the augmented equations form a set of $(b^2+bc)h$ equations in b^2h unknowns.

To make the encryption method secure, the legitimate correspondents must choose $\boldsymbol{R}$, F and M so that these $(b^2+bc)h$ equations are not sufficient to solve for the b^2h unknown entries in R. This is done by making certain that the rank of the commutativity matrix is less than b^2h.

Solving linear equations over a commutative ring is a routine process. Solving linear equations over a non-commutative ring has long been known [10] to be a difficult problem. Many of the familiar properties of matrices no longer hold. There are no eigenvalues or eigenvectors. The value of the determinant changes if columns are added, subtracted or permuted. Canonical decompositions don't work.

The concept of the rank of a matrix is not clearly defined for non-commutative rings. It is not sufficient simply to reduce the matrix to row echelon form and count the rows. To understand the problem better, it is useful to consider the ring of integers modulo 36. The linear equation 5x+7=0 has a unique solution, namely x=13. The equation gives complete information about x. If the equation is multiplied by 2, giving 10x+14=0 then there are 2 solutions, x=13 and x=31. The equation is weaker in the sense that it gives less information about x. If the equation is multiplied by 9 it becomes 9x+27=0, and has 9 solutions. This very weak equation gives much less information about x.

Similarly for linear equations over $\boldsymbol{R}$. Some equations give more information than others, and 2 or 3 equations do not necessarily give more information than a single equation, even when they are linearly independent.

There are several different definitions of

rank in the literature, and it is not always clear how to calculate them. For this study, the rank of the commutativity matrix was defined in terms of the original b^2 unknowns, rather than the b^2h extended unknowns, and seemed to work well. That is, it was a reasonable indicator of the number of solutions to the set of equations.

There is no simple way to characterize a non-commutative ring, therefore there is no simple way to express the conditions that lead to a particular rank for the commutativity matrix of the family F.

In order to obtain numeric results that could be used to estimate how large the matrices need to be, and thus be able to compare running times to the existing exponentiation method, it was decided to construct a ring that was tailored to this encryption scheme. The ring needed to possess a number of desirable properties: the maximum order for a ring element should be as large as possible, there should be as many maximum-order elements as possible, there should be as many invertible elements as possible, and there should be as few commutative elements as possible. The ideal size for the ring would be 256 elements, since that would eliminate any conversion steps from characters of the message to ring elements and back. This gives a ring multiplication table of 65,536 bytes, which is practical even for single-chip devices.

None of the classical examples of non-commutative rings, such as matrices and quaternions, seemed to possess this combination of traits. A computer search produced a large number of suitable 256-element rings that strike various compromises among these conflicting goals. The ring M256 that was chosen has 62 invertible elements, 64 commutative elements, and 31 elements that are both invertible and commutative. It has 30 elements of order 62 and 30 elements of order 31. For the U representation g=7, and for the V representation h=3.

Commutative families F of n×n square matrices over $\boldsymbol{R}$=M256 were generated for n=2 through n=20 and n=29. For n=2 to 9, and for 11 and 13, it is probable that the families have the maximum cardinality. For other values of n it is likely that the families are sub-maximal. Larger families may exist than those which were found. Express the maximum cardinality found as $q(n)^n$. Then q(n) was greater than 100 for n=5 and for n≥8, with the smallest value being 102.01 for n=8 and the largest being 122.35 for n=17.

Subfamilies of these maximal families were generated by using powers of the base matrix. The rank of the commutativity matrix for F was found empirically to be n(n-1) when the cardinality of F evenly divided 32^n-1, and n^2 otherwise. In both cases, for matrices of a practical size it was found that the opponent could solve the equations and recover the message too often to be truly secure.

This was the wrong way to construct the family of matrices.

During the search for matrices of maximal order, thousands of matrices of lower order are generated, tested, and then discarded. It was discovered that when some of these lower-order matrices are used to generate the family F the commutativity equations are not linearly independent, and a large number of trials are required for the opponent to solve them.

This was the right way to construct the family of matrices. Discard both the front-runners and the stragglers, and use the matrices from the middle of the pack.

Suitable lower-order matrices can be found with a small number of trials. It is conjectured that if the lower-order matrix is part of a maximal family, then the equations will be linearly independent, otherwise they will have lower rank, however we know of no way to test this conjecture.

V. Experimental Results

To test the security of this method, and to evaluate how large the matrices must be to achieve the desired level of security, a series of experiments was run. For each matrix size b×b a commutative family of matrices was chosen. 100 matrices were randomly chosen from the family, and used to encrypt a randomly chosen b×1 message block.

The set of linear equations was generated and augmented. Over the non-commutative ring it is not possible to reduce these equations to row echelon form, but an approximation of row echelon form can be obtained using a few simple techniques, such as using pseudo-inverses and looking for invertible linear combinations of elements in the active column. Once the reduced form is obtained, solutions can be obtained by a combination of back substitution and back-tracking.

To illustrate these techniques, recall the ring of integers modulo 36. Suppose one row contains 14x+8y+20=0 and another row contains 6x+11y+3=0. Neither 14 nor 6 is invertible, but multiplying the first row by 15 and adding it to the second row gives 23y+15=0, which eliminates the x term. Similarly, if one row contains 4x+10y+19=0 and another row contains

9x+31y+8=0, neither 4 nor 9 is invertible, but their sum is 13, which is invertible. Here 15 was a pseudo-inverse and 4+9 was a linear combination. These techniques also work in the M256 ring.

During the back substitution, some choices of values will result in impossible conditions at later stages, such as 8x+7=0. This requires back-tracking to resolve the impasse. To keep the execution times of the tests within reasonable bounds, a test case was halted if more than 10^8 sets of values were tried without finding any solution. At that point the program calculated what fraction of the value space had been tried, and then extrapolated the number of trial values needed to search the entire value space. It is understood that searching only 10^8 sets of values in a value space of size 256^{121} or 256^{256} will give only a crude approximation, but by comparing the figures for various matrix sizes a clear trend emerges.

The results of these tests are shown in Table 1. The ranks given are the ranks of the approximate row echelon matrices, stated in terms of the original b^2 unknowns, not the $3b^2$ augmented unknowns. These values can be compared to b^2 to get a sense of how close the equations come to linear independence.

The figures for the work are the logarithms of the estimated number of trials needed, taken to the base 2. That way, the figures can be directly compared to the desired target figure of 128, which is the current standard for security.

For each statistic, the minimum, mean and maximum over 100 trials are given. For the work, it is the minimum figure that is of critical interest, because if that figure is 128 or less, or even close to 128, then an opponent could potentially solve some blocks of the messages, or compute some of the keys. Unless this is a negligible fraction, the method cannot be considered secure. It can be seen from Table 1 that the method begins to become acceptably secure at about 17×17.

Another important consideration for security is the size of the matrix family F. If the family is too small, then the opponent could simply try the matrices sequentially. Recall that the size of the matrix family is 32^b-1, or about 2^{5b}. In order to be secure b should be at least 25 or 26. The value b=29 is suggested because 32^{29}-1 is divisible by a large prime, namely 2679 89515 77838 62814 69002 74941 44991.

The work needed to solve the equations for a 29×29 encryption matrix far exceed the desired level of 2^{128} trials. This suggests that each 29×29 matrix can be used to encrypt more than one 29-character message block. This is of practical importance since the time to generate the encryption matrices is greater than the time to perform the encryption. A new set of experiments was performed to test this possibility.

The results of these experiments are shown in Table 2. It can quickly be seen that it is safe to use each matrix for up to 5 message blocks, but unsafe for 6 message blocks. There is a sharp cutoff. In fact 74% of the 6-block messages were solved.

A few words of explanation may be needed about some seeming oddities in the results. Most of the figures for the minimum work end with . **5**8. At first this was thought to be a consequence of $\log_2 3$=1.58, but investigation showed that it was because $\log_2 10^8$=26.58. It also seemed anomalous that the rank of the equation sets for

Table 1. Work required to solve the equations for an N×1 message block encrypted by an N×N matrix. Size is the encryption matrix size. Rank is the rank of the combined set of equations in terms of the original N^2 variables. Work is $\log_2$ of the estimated number of values which must be tried to obtain a solution. For each measure, the minimum, mean and maximum values are given.

Matrix Size	Rank			Work		
	Min	Mean	Max	Min	Mean	Max
11x11	104	107.55	110	66.58	102.93	111.77
12x12	118	123.91	129	74.58	111.44	118.99
13x13	126	134.62	139	66.58	118.93	130.58
14x14	155	162.37	168	82.58	126.72	135.25
15x15	168	178.34	186	104.01	134.75	146.58
16x16	196	204.96	213	122.58	143.72	151.41
29x29	618	629.75	640	217.58	253.03	483.90

Table 2. Work required to solve the equations for an 29×C message block encrypted by an 29×29 matrix. Size is the message matrix size. Rank is the rank of the combined set of equations in terms of the original N^2 variables. Work is $\log_2$ of the estimated number of values which must be tried to obtain a solution. For each measure, the minimum, mean and maximum values are given.

Message Size	Rank			Work		
	Min	Mean	Max	Min	Mean	Max
29x1	618	629.75	640	217.58	253.03	483.90
29x2	660	678.55	694	210.58	238.29	249.22
29x3	702	725.30	740	178.58	227.31	242.56
29x4	738	768.72	785	178.58	220.40	238.16
29x5	784	806.63	817	184.01	212.12	224.50
29x6	820	839.16	841	16.00	29.29	176.01

the 29×c message blocks increased by more than 29 each time c increased by 1. The reason for this is that in the set of augmented equations each increment of c adds not 29 but 87 additional equations.

Using 29×29 matrices over the M256 ring is just one example, of course, but it demonstrates both the feasibility and practicality of the matrix technique.

VI. Increasing Security

There are several enhancements available to increase the security of the matrix method at a very low computational cost. For a message of n blocks, the number of encryption matrices required by both sender and receiver would be $\lceil n/5 \rceil$. Instead of simply encrypting the first 5 blocks with the first matrix, the next 5 blocks with the second matrix, and so forth, the parties could generate all $\lceil n/5 \rceil$ matrices beforehand, and then choose randomly among them for each block. After a matrix has been used 5 times, it would be discarded.

Since that would require a large amount of storage for a long message, a smaller number of matrices, say 10 to 20, could be generated at the outset. One of these would be chosen at random for each block. After a matrix was used 5 times, it would be replaced by a new matrix. The replacement could stop once $\lceil n/5 \rceil$ matrices had been generated.

A second method is to choose a random multiplier for each block. The multiplier would need to be an invertible and commutative element of the ring. There are 31 such elements in M256. This method can be combined with the previous method.

Conversely, if the parties decided to encrypt the message 5 blocks at a time, the message could be treated as a sequence of larger 29×5 blocks. Each block could be left-multiplied by the 29×29 matrix and right-multiplied by an independent 5×5 matrix chosen from a commutative family G of invertible 5×5 matrices. The right-side matrices are applied and removed the same way as the left side matrices. The 3 transmitted messages would then be SMT, RSMTQ, and S'RSMTQT'=RMQ. Maximal commutative families of 5×5 matrices over M256 contain about 1.4×10^{10} members. The extra cost of generating the 5x5 matrices adds less than 3% to the encryption time and storage requirements.

None of these enhanced methods are needed for key exchange, since only one block is used for each key.

VII. Authentication

Thus far it was assumed that an eavesdropper listened passively. That is, the eavesdropper could read messages, but could not create, alter or delete messages. There is no defense in any system of cryptography against deleted messages, beyond detecting the deletion.

There are many schemes for authenticating that the message you received is from the person you intended [5,6]. No scheme is failure-proof. Every scheme can be defeated by some combination of wiretapping, key-logging, burglary, bribery or coercion. When the opponent controls every channel of access, Internet, LAN, phone, mail, broadcast and even

bonded messenger, there is no defense.

That said, the best way to authenticate messages is through the use of secret information known to sender and receiver. For example, the shared secret information may be combined with a hashed digest of the message in an irreversible way. This violates the spirit of No Key Exchange, but it is the best method available.

VIII. Results

Using the methods just described, two commutative families of 29×29 matrices and 5×5 matrices over a non-commutative ring M256 of 256 elements were constructed, and used to encrypt a message of 5,000,000 characters. The same message was also encrypted and decrypted using a commercial implementation of the Shamir 3-Pass Protocol [1] called NK-Crypt which uses exponentiation modulo a 244-digit prime. The encryption using NK-Crypt took 9 hours 53 minutes, while the matrix encryption took 16.7 seconds, about 2100 times as fast. These times include disk I/O and key generation.

This ratio depends on the size of the prime, the method used for multiplying large numbers and the method of exponentiation, as well as the size of the matrices and the method for generating the key matrices for each block of the message, however, it is safe to say that the new matrix method represents more than a 1000-fold improvement over the prior art.

Until now, private key cryptography using the 3-pass protocol was considered too slow to be used for transmitting anything larger than encryption keys. Since the new matrix method is about 3 orders of magnitude faster, it becomes practical to send entire messages as well as exchanging keys.

Private key encryption using matrices is also at least 1000 times as fast as public key encryption using exponentiation, so whenever speed is important, private key cryptography would now become the preferred method.

References

[1] A. G. Konheim, *Cryptography: A Primer* New York: Wiley, pp. 346-7, 1981.

[2] Letter from Pierre de Fermat to Bernard Frenicle de Bessy, Oct. 18, 1640.

[3] J. L. Massey and J. K. Omura, "Method and apparatus for maintaining the privacy of digital messages conveyed by public transmission," US Patent 4 567 600, Sept. 14, 1982.

[4] N. Courtois, A. Klimov, J. Patarin and A. Shamir, "Efficient Algorithms for Solving Overdefined Systems of Multivariate Polynomial Equations," EUROCRYPT 2000, pp. 392-407.

[5] A. Fiat and A. Shamir, "How to prove yourself: Practical solutions to identification and signature problems," *Advances in Cryptology - Crypto '86*, Springer-Verlag, pp. 186-194, 1987.

[6] T. ElGamal, "A public-key cryptosystem and a signature scheme based on discrete logarithms," IEEE Transactions on Information Theory vol. 31**,** pp. 469-472, 1985.

[7] U. Maurer, "Towards the equivalence of breaking the Diffie-Hellman protocol and computing discrete logarithms," *Advances in Cryptology - Crypto '94*, Springer-Verlag, pp. 271-281, 1994.

[8] W. Diffie and M.E. Hellman, "New directions in cryptography," IEEE Transactions on Information Theory vol. 22, pp. 644-654, 1976.

[9] M. E. Hellman, B. W. Diffie, R. C. Merkle, "Cryptographic apparatus and method," US Patent 4 200 770, Sept. 6, 1977.

[10] O. Ore, "Linear equations in noncommutative rings," Ann. of Math. vol. 32, pp. 463–477, 1931.

Biography

Frank Rubin has done work on printed circuit layout and wiring, text compression, and cryptography. Since retiring from IBM in 1991 he has worked on recreational mathematics, including the knight covering problem, bimagic squares and ABC triples. He is the creator of the Pascal Macro Compiler, and the founder of Master Software Corporation, and also The Contest Center.