Efficient Signature Schemes Based on Birational Permutations

Adi Shamir
Dept. of Applied Mathematics
The Weizmann Institute of Science
Rehovot 76100, Israel
email: shamir@wisdom.weizmann.ac.il

March 1993

Abstract: Many public key cryptographic schemes (such as cubic RSA) are based on low degree polynomials whose inverses are high degree polynomials. These functions are very easy to compute but time consuming to invert even by their legitimate users. To make such schemes more efficient, we consider in this paper the class of birational permutations f over k-tuples of numbers, in which both f and f^{-1} are low degree rational functions. We develop two new families of birational permutations, and show how to transform them into new public key signature schemes which are much faster than the known schemes.

1. Introduction

The original proposal for public key cryptography (Diffie and Hellman [1976] was based on the notion of trapdoor permutations, i.e., invertible functions which are easy to compute but apparently difficult to invert, unless some trapdoor information (which makes the inversion easy) is known. The best known implementation of this idea is the RSA scheme (Rivest, Shamir and Adleman [1978]), which can solve in a unified way the problems of key management, secure transmission, user identification, message authentication, and digital signatures. In one of the variants of this scheme, the encryption function is the low degree polynomial $f(x) = x^3 \pmod{n}$, which can be efficiently computed with two modular multiplications. Unfortunately, its inverse $f^{-1}(v) = v^d \pmod{n}$ is a very high degree polynomial, and thus its evaluation is quite slow (especially in software implementations).

In spite of extensive research in the last 16 years, there had been no fundamentally new constructions of trapdoor permutations which are faster than the

RSA scheme. To overcome this difficulty, researchers have developed specialized solutions to various cryptographic needs which are not based on this unifying notion. For example, Diffie and Hellman [1976] proposed a key management scheme which is based on the one-way permutation of exponentiation modulo a prime. Since this function cannot be efficiently inverted, it is neither an encryption nor a signature scheme. The cryptosystem of Merkle and Hellman [1978] is invertible, but its mapping is not onto and thus it can not generate digital signatures. The Fiat-Shamir [1986] and DSS [1991] signature schemes are not one-to-one mappings, and thus they can not be used as cryptosystems.

A natural approach to this problem is to search for low degree algebraic mappings (polynomials or rational functions) whose inverses are also low degree algebraic mappings. Such mappings are called birational functions. We are particularly interested in multivariate mappings $f(x_1, \ldots, x_k) = (v_1, \ldots, v_k)$ in which the x_i and the v_i are numbers modulo a large $n = pq$, since the solution of general algebraic equations of this type is at least as hard as the factorization of the modulus. In this context, we say that a polynomial modulo n is low degree if its degree is a constant which does not grow with n, and a rational function is low degree if it is the ratio of two low degree polynomials. For example, in the case of cubic RSA, the function is considered low degree, but its inverse is not. Non-linear algebraic mappings do not usually have unique inverses, when they do have inverses they usually cannot be written in closed form, and when the closed forms exist they are usually based on root extractions (radicals) or exponentiations whose computation modulo a large n is very slow. The construction of non-linear birational mappings is thus a non-trivial task.

One attempt to construct birational permutations, due to Fell and Diffie [1985], used the following DES-like idea. Let $(x_1, x_2, \ldots, x_k)$ be an initial k-vector of variables, and let g be a secret multivariate polynomial. Alternately replace the current k-vector of multivariate polynomials $(p_1, p_2, \ldots, p_k)$ by $(p_1 + g(p_2, \ldots, p_k), p_2, \ldots, p_k)$, and rotate the k-vector to the right. After sufficiently many iterations, expand and publish the resultant k-vector of multivariate polynomials as your public key. When the trapdoor information g is known, the inverse of f can be computed by undoing the transformations (i.e., by alternately subtracting $g(p_2, \ldots, p_k)$ from p_1 and rotating the k-vector to the left). Unfortunately, even when g is a quadratic function, the number of terms can grow double exponentially with the number of iterations, which cannot be too small for security reasons. As the authors themselves conclude, "there seems to be no way to build such a system that is both secure and has a public key of practical size".

The problem is not accidental, due to the following generic attack: If f is known, the cryptanalyst can prepare a large number of input-output pairs for this function. Since f is invertible, these pairs (in reverse order) can be used to interpolate the unknown low-degree function f^{-1} by solving a small number of linear equations relating its coefficients. We do not know how to avoid this problem in the context of public key cryptosystems. However, in the context of

public key signature schemes we propose a simple modification which overcomes the interpolation attack. It has the following general form:

Key Generation: Each user in the system chooses a particular birational permutation $f(x_1,\ldots,x_k)=(v_1,\ldots,v_k)$ consisting of k rational functions $f_i(x_1,\ldots,x_k) = v_i$, discards the first $s>0$ of these f_i functions, describes the other $k-s$ f_i functions in his public key, and keeps the inverse of f as his private key.

Signature Generation: Given a digital message m, the signer chooses $v_i = r_i$ for $i = 1,\ldots,s$, and computes $v_i = h(m,i)$ for $i = s+1,\ldots,k$, where r_i are newly chosen secret random values, and h is a publicly known cryptographic hash function. He then uses his knowledge of the secret f^{-1} to compute a signature $(x_1,\ldots,x_k)$ satisfying $f(x_1,\ldots,x_k) = (v_1,\ldots,v_k)$.

Signature Verification: The verifier checks that $f_i(x_1,\ldots,x_k) = h(m,i)$ for $i = s+1,\ldots,k$, where the f_i's are taken from the signer's public key.

The scheme cannot be used as a cryptosystem, since the cleartext $(x_1,\ldots,x_k)$ cannot be uniquely recovered from the shorter ciphertext $(v_{s+1},\ldots,v_k)$. It can be used as a signature scheme, since messages can have multiple signatures. The cryptanalyst cannot interpolate f^{-1} since it is not uniquely defined by the public key: He cannot generate complete input-output pairs for f^{-1} by himself, and cannot use input-output pairs generated by the legitimate signer since each one of them is based on new unknown values r_i. The recommended choice of s is 1, which makes the verification condition hardest to satisfy.

The security of this scheme depends on the choice of birational permutations. In Section 2 we introduce a simple family of birational permutations based on sequentially linearized equations, and in Section 3 we introduce a more sophisticated family of birational permutations based on algebraic bases in polynomial rings. The two families yield new signature schemes which seem to combine high security with very low computational complexity.

2. A Signature Scheme Based on Sequentially Linearized Equations

Let n be the public product of two large secret primes p and q. Consider the triangular birational permutation $g(y_1,\ldots,y_k) = (w_1,\ldots,w_k) \pmod n$ in which the i-th output depends only on the first i inputs via the mapping $g_i(y_1,...,y_i) = w_i \pmod n$, where g_i is a low degree polynomial which is linear in its last input y_i (the other inputs can and should occur non-linearly). Given the values of inputs $y_1,\ldots,y_k$, we can easily compute the values of the outputs $w_1,\ldots,w_k$ by evaluating k low degree polynomials. Given the values of the outputs $w_1,\ldots,w_k$, we can easily recover the values of the inputs $y_1,\ldots,y_k$ by solving a series of linear equations: First we solve for y_1 in $g_1(y_1) = w_1 \pmod n$. Then we substitute the computed value of y_1 into its (non-linear) occurrences in $g_2(y_1,y_2) = w_2 \pmod n$, and solve the remaining linear equation in y_2. We proceed in this order until we

compute the last y_k. Each y_i is thus a low degree rational function of the w_j's, which is easy to compute with a small number of arithmetic operations modulo n.

To hide the easy solvability of the g_i's, the user has to transform them into more random looking polynomials before publishing them as his public key. We recommend the following two transformations:

1. Let A be a randomly chosen invertible $k \times k$ matrix, and consider the variable transformation $Y = AX$, where Y is the column vector of original variables $(y_1, \ldots, y_k)^t$ and X is a column vector of new variables $(x_1, \ldots, x_k)^t$. When the resultant polynomials are expanded, they contain all the variables in a non-linear way.

2. Let B be a randomly chosen invertible $k \times k$ matrix, and consider the mixing transformation $F = BG$, where G is the column vector of polynomials $(g_1, \ldots, g_k)^t$ and F is a column vector of new polynomials $(f_1, \ldots, f_k)^t$. Each f_i is thus a polynomial whose coefficients are random linear combinations of the corresponding coefficients of the given $g_1, \ldots, g_k$.

When A and B are known, it is easy to solve the resultant system of equations $f_i(x_1, ..., x_k) = v_i \pmod{n}$ for $i = 1, ..., k$ by inverting these transformations. To minimize the size of the public key, we recommend using g_i's which are homogeneous quadratic expressions of the form:

$$g_i(y_1, \ldots, y_i) = l_i(y_1, \ldots, y_{i-1}) \cdot y_i + q_i(y_1, \ldots, y_{i-1}) \pmod{n}$$

where l_i is a randomly chosen linear function of its inputs and q_i is a randomly chosen homogeneous quadratic function of its inputs. The only exception is g_1 in which l_1 and q_1 have no inputs. Since the coefficients of the linear g_1 cannot be mixed with the coefficients of the quadratic $g_2, \ldots, g_k$, and since we have to eliminate at least one of the polynomials in order to overcome the interpolation attack, we recommend the elimination of g_1 from the user's public key.

Without loss of generality, we can assume that $g_1(y_1) = y_1$ and $g_2(y_1, y_2) = y_1 y_2$ (since they can always be brought to this form by linear transformations). The case $k = 2$ is thus equivalent to the OSS scheme (Ong, Schnorr, Shamir [1983]), where the variable transformation A is $y_1 = x_1 + ax_2$, $y_2 = x_1 - ax_2$ and the mixing transformation B is the identity (all the arithmetic operations are carried out modulo a composite n). The OSS scheme was successfully attacked by Pollard [1984]. However, his attack is specific to one quadratic or cubic equation in several variables, and does not seem to apply to systems of two or more simultaneous equations. The minimum recommended value of k is thus 3, but further analysis may change it to 4 or some higher value. A typical example of the new signature scheme for $k = 3$ with the toy modulus $n = 101$ is:

Example: Consider the following sequentially linearized system of equations:

$$\begin{aligned} y_1 &= w_1 \pmod{101} \\ y_1 y_2 &= w_2 \pmod{101} \\ (29y_1 + 43y_2)y_3 + (71y_1^2 + 53y_2^2 + 89y_1 y_2) &= w_3 \pmod{101}. \end{aligned}$$

Apply the linear change of variables:

$$\begin{aligned} y_1 &= x_1 + 25x_2 + 73x_3 \pmod{101} \\ y_2 &= x_1 + 47x_2 + 11x_3 \pmod{101} \\ y_3 &= x_1 + 83x_2 + 17x_3 \pmod{101} \end{aligned}$$

to obtain the new expressions:

$$\begin{aligned} x_1 + 25x_2 + 73x_3 &= w_1 \pmod{101} \\ x_1^2 + 64x_2^2 + 96x_3^2 + 72x_1x_2 + 84x_1x_3 + 70x_2x_3 &= w_2 \pmod{101} \\ 83x_1^2 + 55x_2^2 + 16x_3^2 + 28x_1x_2 + 97x_1x_3 + 74x_2x_3 &= w_3 \pmod{101}. \end{aligned}$$

Mix these three expressions g_1, g_2, and g_3 by computing: $f_1 = g_1 \pmod{101}$, $f_2 = (39g_2 + 82g3) \pmod{101}$, $f_3 = (93g_2 + 51g3) \pmod{101}$. The resultant expressions are:

$$\begin{aligned} x_1 + 25x_2 + 73x_3 &= v_1 \pmod{101} \\ 78x_1^2 + 37x_2^2 + 6x_3^2 + 54x_1x_2 + 19x_1x_3 + 11x_2x_3 &= v_2 \pmod{101} \\ 84x_1^2 + 71x_2^2 + 48x_3^2 + 44x_1x_2 + 33x_1x_3 + 83x_2x_3 &= v_3 \pmod{101}. \end{aligned}$$

Discard the first equation, publish the last two equations as your public key, and retain the variable transformation A and the mixing transformation B as your secret key.

To sign a message m, hash it into two numbers v_2, v_3 in $[0, 101)$. Assume that $v_2 = h(m, 2) = 12$ and $v_3 = h(m, 3) = 34$. Its signature is any triplet of numbers x_1, x_2, x_3 which satisfies the two published equations for these values of v_2, v_3. To find such a solution, choose a random value for v_1 (each choice will yield a different signature). For example, let $v_1 = 99$. Apply the inverse mixing transformation B^{-1} by computing $w_1 = v_1 = 99 \pmod{101}, w_2 = 8v_2 + v_3 = 29 \pmod{101}, w_3 = 27v_2 + 18v_3 = 27 \pmod{101}$. The equations in terms of the y_i's are thus:

$$\begin{aligned} y_1 &= 99 \pmod{101} \\ y_1 y_2 &= 29 \pmod{101} \\ (29y_1 + 43y_2)y_3 + (71y_1^2 + 53y_2^2 + 89y_1 y_2) &= 27 \pmod{101}. \end{aligned}$$

To solve it, substitute $y_1 = 99$ into the second equation and solve the linear equation $99y_2 = 29 \pmod{101}$ to obtain $y_2 = 36$. Substitute both values into the third equation, simplify it into $76y_3 + 45 = 27 \pmod{101}$, and solve it to obtain $y_3 = 29$. Finally, use the inverse of the variable transformation A to change this Y solution into the X solution $x_1 = 40, x_2 = 27, x_3 = 22$ which is a valid signature of m since it satisfies the two published equations. ∎

Two general homogeneous quadratic equations in three unknowns have 12 coefficients, but they can be changed without loss of generality into a normal form which reduces the size of the public key to five numbers. The size of each signature is 196 bytes (when n is a 512-bit modulus). The generation and verification of each signature requires about 20 modular multiplications, compared to about 760 modular multiplications in the RSA scheme and about 50 modular multiplications in the Fiat-Shamir Scheme. These parameters will be somewhat larger if we prefer to use the less marginal $k = 4$.

We can formally prove that the trapdoor information cannot be successfully recovered by analyzing the public key. To do so, we use the new technique introduced in Shamir [1993] to prove that any algorithm for computing A and B from the public key is at least as difficult as the factorization of the modulus. This technique is applicable whenever the problem remains invariant under some non-trivial algebraic transformation, and in our case we use the symmetry of the problem when the definitions of y_1 and y_2 are exchanged in A. However, this result does not rule out Pollard-like attacks, which solve the given equations directly without finding the original trapdoor information. More details on the provable and unprovable aspects of the security of the new scheme will be given in the full version of this paper.

3. A Signature Scheme Based On Algebraic Bases

In this section we introduce a different family of birational permutations, which is inherently non-linear in all its components. The construction is quite unexpected, and seems to introduce powerful new techniques for constructing invertible mappings which can have other cryptographic applications.

Let $F_d[y_1, y_2, \ldots, y_k]$ denote the set of all the homogeneous polynomials of degree d in the k variables $y_1, y_2, \ldots, y_k$. We consider in particular the case of quadratic polynomials ($d = 2$), which are linear combinations of the $k(k+1)/2$ elementary quadratics $y_i y_j$ for $i \leq j$ with coefficients a_{ij}. Such a set of quadratics is called a linear basis of $F_2[y_1, \ldots, y_k]$.

If we allow the additional operations of multiplication and (remainder-free) division of polynomials, we can express some of the elementary quadratics by other elementary quadratics. For example, the standard linear basis of $F_2[y_1, y_2, y_3]$ is $\{y_1^2, y_2^2, y_3^2, y_1y_2, y_1y_3, y_2y_3\}$. However, the first three elementary quadratics can

be expressed by the last three in the following way:

$$y_1^2 = (y_1y_2)(y_1y_3)/(y_2y_3),\ y_2^2 = (y_1y_2)(y_2y_3)/(y_1y_3),\ y_3^2 = (y_1y_3)(y_2y_3)/(y_1y_2).$$

We can thus reduce the six linear generators $\{y_1^2, y_2^2, y_3^2, y_1y_2, y_1y_3, y_2y_3\}$ into the three algebraic generators $\{y_1y_2, y_1y_3, y_2y_3\}$. Another triplet of algebraic generators is $\{y_1^2, y_1y_2, y_2y_3\}$. However, $\{y_1^2, y_2^2, y_2y_3\}$ does not algebraically generate $F_2[y_1, y_2, y_3]$ since it cannot express y_1y_2.

To formalize this notion, consider an arbitrary set G of polynomials over a ring R. Its algebraic closure $[G]$ of G is defined as the smallest set of polynomials which contains G and R, and is closed under addition, subtraction, multiplication, and remainder-free division. Note that $[G]$ is not necessarily an ideal in the ring of polynomials over R, since we do not allow arbitrary polynomials as coefficients of the generators. For example, when $G = \{y^2\}$, $[G]$ is the set of all the polynomials in y whose monomials have even degrees. Note further that root extractions (radicals) are not allowed as basic operations, since they cannot be carried out efficiently in some rings R. G is called an algebraic basis of F if the polynomials in G are algebraically independent and F is contained in $[G]$.

Theorem: *1. The set of polynomials* $\{y_1^2, y_1y_2, y_2y_3, \ldots, y_{k-1}y_k\}$ *is an algebraic basis of* $F_2[y_1, y_2, \ldots, y_k]$ *for any* k.

2. The set of polynomials $\{y_1y_2, y_2y_3, y_3y_4, \ldots, y_ky_1\}$ *is an algebraic basis of* $F_2[y_1, y_2, \ldots y_k]$ *for any odd* $k > 1$.

Proof (sketch): We show that the specified sets can generate all the other elementary quadratics y_iy_j for $i \leq j$. Assume first that $j - i$ is an odd number. Then we can use the telescoping formula:

$$y_iy_j = (y_iy_{i+1})(y_{i+2}y_{i+3})\ldots(y_{j-1}y_j)/(y_{i+1}y_{i+2})\ldots(y_{j-2}y_{j-1}).$$

If $j - i$ is an even number, this approach will yield y_i/y_j instead of y_iy_j. To turn the former into the later, we have to multiply it by y_j^2. In case 1, we are given y_1^2 as a generator, and we can turn each y_t^2 into y_{t+1}^2 by using the formula:

$$y_{t+1}^2 = (y_ty_{t+1})^2/y_t^2.$$

In case 2, we use the fact that k is odd, and therefore the distance from j to j via the cyclic order $y_j, y_{j+1}, \ldots, y_k, y_1, \ldots, y_j$ is odd. The telescoping formula will thus yield the desired y_j^2 if we cycle through all the variables y_t. ∎

The Theorem specifies two types of algebraic bases of size k for $F_2[y_1, \ldots, y_k]$. To get other bases, we can use the variable transformation A and the mixing transformation B described in Section 2. Due to the invertible and algebraic

nature of these transformations, it is easy to translate the representation of each f in the original basis into a representation of f in the new bases.

Example: Consider the algebraic basis $G = \{y_1y_2, y_2y_3, y_3y_1\}$ of $F_2[y_1, y_2, y_3]$, and the randomly chosen linear transformations:

$$A = \begin{pmatrix} 37 & 62 & 71 \\ 89 & 45 & 68 \\ 50 & 17 & 93 \end{pmatrix} \qquad B = \begin{pmatrix} 41 & 73 & 51 \\ 89 & 12 & 60 \\ 37 & 94 & 19 \end{pmatrix}$$

Then the linear change of variables $Y \leftarrow AY \pmod{101}$ changes the generators g_i in G into:

$$\begin{aligned} g'_1 &= (37y_1 + 62y_2 + 71y_3)(89y_1 + 45y_2 + 68y_3) \\ &= 12y_1y_2 + 38y_2y_3 + 48y_3y_1 + 61y_1^2 + 63y_2^2 + 81y_3^2 \pmod{101} \\ g'_2 &= (89y_1 + 45y_2 + 68y_3)(50y_1 + 17y_2 + 93y_3) \\ &= 26y_1y2 + 89y_2y_3 + 62y_3y_1 + 6y_1^2 + 58y_2^2 + 62y_3^2 \pmod{101} \\ g'_3 &= (50y_1 + 17y_2 + 93y_3)(37y_1 + 62y_2 + 71y_3) \\ &= 93y_1y_2 + 4y_2y_3 + 22y_3y_1 + 32y_1^2 + 44y_2^2 + 38y_3^2 \pmod{101} \end{aligned}$$

and the linear transformation $G'' \leftarrow BG' \pmod{101}$ changes the generators g'_i in G' into the new generators:

$$\begin{aligned} g''_1 &= 41g'_1 + 73g'_2 + 51g'_3 \\ &= 63y_1y_2 + 78y_2y_3 + 41y_3y_1 + 26y_1^2 + 72y_2^2 + 89y_3^2 \pmod{101} \\ g''_2 &= 89g'_1 + 12g'_2 + 60g'_3 \\ &= 92y_1y_2 + 44y_2y_3 + 74y_3y_1 + 48y_1^2 + 55y_2^2 + 32y_3^2 \pmod{101} \\ g''_3 &= 37g'_1 + 94g'_2 + 19g'_3 \\ &= 9y_1y_2 + 51y_2y_3 + 43y_3y_1 + 96y_1^2 + 34y_2^2 + 53y_3^2 \pmod{101} \end{aligned}$$

In the original basis G it was easy to find the representation of any given quadratic polynomial f as an algebraic expression in the g_i's. In a transformed basis G'' it is not clear how to find such a representation, unless the invertible transformations are known. ∎

To simplify our notation, we assume without loss of generality that $k > 1$ is odd and G is always the symmetric basis $\{y_iy_{i+1}\}$ (where $i + 1$ is computed mod k). Given the invertible linear transformation $Y \leftarrow AY$, we define $A\&A$ as the $k \times k(k+1)/2$ matrix whose i-th row represents the quadratic polynomial y_iy_{i+1} after the change of variables. The coefficients of the final basis G'' can thus be compactly represented by the $k \times k(k+1)/2$ matrix $B(A\&A)$.

Assume now that we are given an arbitrary assignment of values $x_1, \ldots, x_k$ to the elementary quadratics y_1y_2, $y_2y_3, \ldots, y_ky_1$ in the standard basis of $F_2[y_1, \ldots,$

$y_k]$ (we use the algebraic independence of these generators to make any such assignment consistent, without actually finding the values of the individual y_i's). We can use the telescoping formulas to compute the values of all the $k(k+1)/2$ elementary quadratics $y_i y_j$ for $i \leq j$. We denote this extended version of X by $E(X)$, and note that it increases the height of the column vector from k to $k(k+1)/2$. The values of the k generators in G'' for this assignment X can be computed by the matrix-vector product $V = [B(A\&A)][E(X)]$. Note again the non-linearity of this transformation, which is due to the quadratic computation of the coefficients of A&A from the coefficients of A, and the multiplications and divisions required to extend X into $E(X)$.

This function maps the k-vector X into the k-vector V. Our goal now is to invert this function and recover the original values in X when A and B are known. First, we can undo the effect of B by computing $W = B^{-1}V$, and obtain the relationship $W = [A\&A][E(X)]$. By definition, the values w_i in W are the values of the generators g'_i in the intermediate G'. Since G' is an algebraic basis, it can represent any quadratic polynomial in $F_2[y_1, \ldots, y_k]$ as an algebraic expression in the generators g'_i. In particular it can represent the k elementary quadratics $y_i y_{i+1}$ in such a way. However, each x_i was defined as the value of $y_i y_{i+1}$, and thus it can be recovered by evaluating an appropriate algebraic expression in the values $w_1, w_2, \ldots, w_k$. It is easy to check that these algebraic expressions can be compactly represented as $X = [A^{-1}\&A^{-1}][E(W)]$.

Example (continued): Consider the algebraic bases G, G' and G'' of the previous example. Let x_i denote the value of $y_i y_{i+1}$, i.e.: $y_1 y_2 = x_1$, $y_2 y_3 = x_2$, $y_3 y_1 = x_3 \pmod{101}$. The values of the other three elementary quadratics can be expressed by as $y_1^2 = x_3 x_1 / x_2$, $y_2^2 = x_1 x_2 / x_3$, $y_3^2 = x_2 x_3 / x_1 \pmod{101}$. The values v_i of the g''_i are computed via $[B(A\&A)][E(X)]$, i.e.:

$$v_1 = 63x_1 + 78x_2 + 41x_3 + 26x_3x_1/x_2 + 72x_1x_2/x_3 + 89x_2x_3/x_1 \pmod{101}$$
$$v_2 = 92x_1 + 44x_2 + 74x_3 + 48x_3x_1/x_2 + 55x_1x_2/x_3 + 32x_2x_3/x_1 \pmod{101}$$
$$v_3 = 9x_1 + 51x_2 + 43x_3 + 96x_3x_1/x_2 + 34x_1x_2/x_3 + 53x_2x_3/x_1 \pmod{101}$$

In particular, when the input is $x_1 = 1$, $x_2 = 2$, $x_3 = 3$, the output is $v_1 = 54$, $v_2 = 63$, $v_3 = 85$. To invert this mapping, we reverse the transformation $G'' \leftarrow BG' \pmod{101}$ by computing $W = B^{-1}V \pmod{101}$. When $v_1 = 54$, $v_2 = 63$, $v_3 = 85$, this matrix-vector product yields $w_1 = 94$, $w_2 = 69$, $w_3 = 1$, which are the values of the intermediate generators g'_1, g'_2, g'_3. We then extend this W into $E(W) = (w_1, w_2, w_3, w_3w_1/w_2, w_1w_2/w_3, w_2w_3/w_1)$, and compute $X = [A^{-1}\&A^{-1}][E(W)] \pmod{n}$, i.e.:

$$x_1 = 29w_1 + 75w_2 + 45w_3 + 74w_3w_1/w_2 + 42w_1w_2/w_3 + 45w_2w_3/w_1 \pmod{101}$$
$$x_2 = 46w_1 + 72w_2 + 14w_3 + 99w_3w_1/w_2 + 61w_1w_2/w_3 + 58w_2w_3/w_1 \pmod{101}$$
$$x_3 = 1w_1 + 58w_2 + 46w_3 + 87w_3w_1/w_2 + 31w_1w_2/w_3 + 77w_2w_3/w_1 \pmod{101}.$$

For $w_1 = 94$, $w_2 = 69$, $w_3 = 1$, the extension of W to $E(W)$ yields the column vector $(94, 69, 1, 16, 22, 19)^t$. When these values are substituted into the expressions above, we get the original inputs $x_1 = 1$, $x_2 = 2$, $x_3 = 3$. ∎

Note that the application and inversion of the function have essentially the same form. They use a small number of modular additions multiplications and divisions but no root extractions.

The new signature scheme can now be formalized in the following way:

Key Generation: Pick a set F of rational functions in k variables, and a standard algebraic basis G of F with the property that the representation of any f in F as an algebraic expression in terms of the generators g_i in G can be easily computed. Transform the easy basis G into an apparently hard basis G'' by using randomly chosen invertible algebraic transformations. Eliminate $s > 0$ generators from G'', publish the other $k - s$ generators in G'' as the public key, and keep the algebraic transformations as the private key.

Signature Generation: To sign a given message m, assign to each published g_i'' the hashed value $v_i = h(m, i)$ of m, and to each eliminated g_i'' a newly chosen random number r_i. Use the secret algebraic transformations to express each g_i in the easy basis G in terms of the generators g_j'' in the hard basis G''. The values x_i of the easy generators g_i for this assignment of values to the hard generators g_j'' form the signature X of m.

Signature Verification: Assign the values x_i from the signature X to the easy generators g_i, and compute the values v_i of the $k - s$ hard generators $g''i$ which appear in the signer's public key. Evaluate the $k - s$ hashed forms $h(m, i)$ of m, and accept the validity of the signature if $v_i = h(m, i)$ for all the $k - s$ values of i.

The recommended choice for F is the set $F_d[y_1, \ldots, y_k]$ of all the homogeneous polynomials of degree d over the ring Z_n. The factorization of $n = pq$ can be destroyed as soon as n is chosen and published. The recommended choice for G is some set of monomials $y_1^{e_1} y_2^{e_2} \ldots y_k^{e_k}$ with $e_1 + e_2 + \ldots + e_k = d$ such that any other monomial in $F_d[y_1, \ldots, y_k]$ can be generated by a sequence of multiplications and divisions. It is not difficult to show that for any d and k, $F_d[y_1, \ldots, y_k]$ has an algebraic basis of this type which consists of exactly k monomials of degree d (for example, $G = \{y_1^3, y_2^2 y_3, y_1 y_2 y_3\}$ is an algebraic basis for $F_3[y_1, y_2, y_3]$). The problem with large choices of k and d is that the number of coefficients in the published generators G'' grows as $O(k^{d+1})$. However, for fixed d this key size grows only polynomially in k, and the recommended choice of d is 2. The recommended bases for $d = 2$ are the standard bases $G = \{y_1 y_2, \ldots, y_{k-1} y_k, y_k y_1\}$ for odd k and $G = \{y_1^2, y_1 y_2, \ldots, y_{k-1} y_k\}$ for arbitrary k. The minimum recommended value of k is currently $k = 3$, and the recommended choice of invertible algebraic transformations is a pair of randomly chosen $k \times k$ matrices A (the variable transformation) and B (the mixing transformation). Note that bases G with $\mathrm{GCD}(g_1, g_2, \ldots, g_k) \neq 1$ should be avoided — such a GCD remains nontrivial under the transformations $Y \leftarrow AY$ and $G'' \leftarrow BG'$, and its computation can reveal

some of the rows of A. This condition is automatically satisfied in our standard bases G for $k \geq 3$. For example, when $k = 3$, two random linear combinations of $y_1 y_2$, $y_2 y_3$ and $y_3 y_1$ will almost surely be relatively prime. This signature scheme is comparable in its key size, signature size, and computational complexity to the signature scheme described in Section 2. We can again prove some partial results about the security of the new birational permutation:

Theorem: *Any algorithm for recovering the secret transformations A, B from the public generators defined by $K = B(A\&A) \pmod n$ or for recovering $y_1, \ldots, y_k$ from $v_1, \ldots, v_k$ can be used to factor n with the same complexity. This remains true even when the public key consists of the first $k - s$ rows in K. However, if K and one of A, B is known, the other can be easily computed.*

The theorem proves that the recovery of the original trapdoor is difficult, but it does not rule out other attacks. It also explains why we use two transformations — if we do not apply either A or B the scheme becomes insecure. Further details about the security and efficiency of the proposed scheme will appear in the full version of this paper.

Bibliography

1. W. Diffie and M. Hellman [1976]: "New Directions in Cryptography", IEEE Trans. Information Theory, Vol IT-22, No 6, pp 644-654.
2. DSS [1991]: "Specifications for a Digital Signature Standard", US Federal Register Vol 56 No 169, August 30 1991.
3. H. Fell and W. Diffie [1985]: "Analysis of a Public Key Approach Based on Polynomial Substitution", Proceedings of CRYPTO 85, Springer-Verlag Vol 218, pp 340-349.
4. A. Fiat and A. Shamir [1986]: "How to Prove Yourself: Practical Solutions to Identification and Signature Problems", Proceedings of Crypto 86, Springer-Verlag Vol 263, pp 186-194.
5. R. Merkle and M. Hellman [1978]: "Hiding Information and Signatures in Trapdoor Knapsacks", IEEE Trans. Information Theory, Vol IT-24, No 5, pp 525-530.
6. R. Rivest, A. Shamir and L. Adleman [1978]: "A Method for Obtaining Digital Signatures and Public Key Cryptosystems", Comm. ACM, Vol 21. No 2, pp 120-126.
7. A. Shamir [1993]: "On the Generation of Multivariate Polynomials Which Are Hard To Factor", Proceedings of STOC 1993.