METHOD OF PERFORMING ELLIPTIC POLYNOMIAL CRYPTOGRAPHY WITH ELLIPTIC POLYNOMIAL HOPPING

Inventors: Lahouari Ghoufi, Dhahran (SA); Mohammad K. Ibrahim, Leicester (GB); Adnan A. Gutub, Dhahran (SA)

Assignee: King Fahd University of Petroleum and Minerals, Dhahran (SA)

Notice: Subject to any disclaimer, the term of this patent is extended or adjusted under 35 U.S.C. 154(b) by 527 days.

Appl. No.: 12/656,899
Filed: Feb. 18, 2010

Prior Publication Data
US 2011/0200185 A1 Aug. 18, 2011

Int. Cl.
H04L 9/28 (2006.01) H04L 9/30 (2006.01) H04K 1/02 (2006.01)

U.S. Cl. 380/28; 380/28; 380/30; 380/43; 380/46

Field of Classification Search 380/28, 380/30, 43, 46

See application file for complete search history.

References Cited
U.S. PATENT DOCUMENTS
4,668,103 A 5/1987 Wilson
4,995,062 A 2/1991 Schonherr
5,054,066 A 10/1991 Rick et al.
5,146,500 A 9/1992 Maurer
5,150,411 A 9/1992 Maurer
5,272,755 A 12/1993 Miyaji et al.

FOREIGN PATENT DOCUMENTS
EP 0892520 1/1999

OTHER PUBLICATIONS
Chevassut, Olivier et al. “The Twist-AUGmented Technique for Key Exchange”, 2006 *

Primary Examiner — Michael Simitoski
Attorney, Agent, or Firm — Richard C. Litman

ABSTRACT
The method of performing elliptic polynomial cryptography with elliptic polynomial hopping allows for the encryption of messages through elliptic polynomial cryptography, i.e., using elliptic polynomials with multiple variables, and particularly with the utilization of elliptic polynomial hopping based upon both the elliptic polynomial and its twist, regardless of whether the elliptic polynomial and its twist are isomorphic with respect to one another. Each plaintext block is encrypted by a different elliptic polynomial, and the elliptic polynomials used are selected by an initial secret key and a random number generator. The method is particularly useful for symmetric encryption systems, and provides a block cipher fundamentally based upon a computationally hard problem.

3 Claims, 1 Drawing Sheet
### U.S. PATENT DOCUMENTS

<table>
<thead>
<tr>
<th>Patent Number</th>
<th>Date</th>
<th>Inventor(s)</th>
</tr>
</thead>
<tbody>
<tr>
<td>7,483,533 B2</td>
<td>1/2009</td>
<td>Ibrahim</td>
</tr>
<tr>
<td>7,483,534 B2</td>
<td>1/2009</td>
<td>Ibrahim</td>
</tr>
<tr>
<td>2006/0029220 A1</td>
<td>2/2006</td>
<td>Ibrahim</td>
</tr>
<tr>
<td>2006/0029221 A1</td>
<td>2/2006</td>
<td>Ibrahim</td>
</tr>
</tbody>
</table>

### FOREIGN PATENT DOCUMENTS

<table>
<thead>
<tr>
<th>Country</th>
<th>Patent Number</th>
<th>Date</th>
</tr>
</thead>
<tbody>
<tr>
<td>WO</td>
<td>WO 9904332</td>
<td>1/1999</td>
</tr>
</tbody>
</table>

### OTHER PUBLICATIONS


* cited by examiner
METHOD OF PERFORMING ELLIPTIC POLYNOMIAL CRYPTOGRAPHY WITH ELLIPTIC POLYNOMIAL HOPPING

BACKGROUND OF THE INVENTION

1. Field of the Invention

The present invention relates to computerized cryptographic methods for communications in a computer network or electronic communications system, and particularly to a method of performing elliptic polynomial cryptography with elliptic polynomial hopping.

2. Description of the Related Art

Encryption systems have been categorized in the art into two types, based on the characteristics of the cryptographic key: asymmetric encryption systems and symmetric encryption systems. Asymmetric encryption systems are also called dual-key systems or public-key systems. In the public-key encryption systems, there is a public key and a private key. The public key is made public, while the private key remains secret. Encryption is performed with the public key, while decryption is done with the private key. The public key and the private key are totally different, and it is difficult to derive one by inference from the other. Thus, the ability to encrypt can be separated from the ability to decrypt.

Symmetric encryption systems are also called single-key systems, private-key systems, or traditional encryption systems. Private-key systems can be categorized into stream encryption systems and block encryption systems. In a stream encryption system, plain data is encrypted on an element-by-element basis (in computer systems, one letter, character, or bit at a time). In contrast, in a block encryption system, plain data is divided into blocks with each block containing plural elements, and encryption is carried out on a block-by-block basis (in computer systems, blocks of letters, characters, or bits at a time).

Block ciphers are the most popular algorithms in use today for providing data privacy. Block ciphers with block size n and key size k can be viewed as a family of permutations on the set of all n-bit strings, indexed by k-bit long encryption keys and possessing certain properties. Block ciphers use fix-sized blocks. A 128-bit plain text block will generate a 128-bit ciphertext block when encrypted. If the message is shorter than the block size, the message is padded to bring the block size up to whatever is required by the particular block cipher being used. If the message is longer than the block size, then the message will be encoded using a mode or method designed to make the encrypted message less vulnerable to attack.

From the mid-1970s through the 1990s, the Data Encryption Standard (DES) was probably the most popular and widely used block cipher cryptographic algorithm. DES had a block size of 64 bits and a key size of 56 bits, and implemented a Feistel structure built on a series of exclusive-or (XOR) operations. When the message size exceeded the block size, DES employed one of several modes. The simplest, electronic codebook (ECB), is too vulnerable to attack if the message is too long and has repeating phrases because two blocks of equivalent plaintext characters always encrypt to the same block of ciphertext. Other modes include cipher block chaining (CBC), cipher feedback (CFB), and output feedback (OFB), of which CBC is the most popular. In CBC, each plaintext block is XORed with the previous plaintext block, beginning with an initialization vector for the first block.

Once thought secure, DES has proved vulnerable to attack through increases in the speed and memory storage of computer hardware. It is now possible for an attacker to decrypt a DES encrypted message by an exhaustive key search in a comparatively short amount of time. Although DES, or variations of DES, such as 3DES or Triple DES, is still used, it is being replaced by the Advance Encryption Standard (AES).

AES uses a 128-bit block size and may use a key size of 128 bits, 192 bits, or 256 bits. The algorithm has a Rijndael structure, which again relies upon a series of XOR operations. Nevertheless, there are concerns about the adequacy of the AES algorithm due to its simple algebraic structure. Moreover, the increase in block size may be inconvenient to implement, is not truly scalable (only three key sizes are implemented or permitted), the hardware for a 192-bit key is not completely compatible with the hardware for a 128-bit or 256-bit key; and advances in computer hardware speed and memory may eventually make AES vulnerable to an exhaustive key search in the same manner as DES.

The fundamental problem with DES, AES, and other similar block cipher algorithms is that these algorithms are essentially based upon simple XOR operations, and not upon a computationally hard or infeasible problem, such as the discrete logarithm problem or the elliptic curve (EC) analog to the discrete logarithm problem.

Elliptic curve cryptosystems are well known in the art. Elliptic curve cryptosystems rely upon the difficulty of the Elliptic Curve Discrete Logarithm Problem (ECDLP) to provide their effectiveness as a cryptosystem. Using multiplicative notation, the problem can be described as: given points B and Q in the group (points in a finite field satisfying a given elliptic curve equation), find a number k such that B^k = Q; where k is called the discrete logarithm of Q to the base B. Using additive notation, the problem becomes: given two points B and Q in the group, find a number k such that kB = Q.

Conventional elliptic curve arithmetic and properties, including point addition, scalar multiplication, point doubling, projective coordinates, etc., are well known in the art, and have been described in numerous texts, treatises, and cryptography handbooks, e.g., Guide to Elliptic Curve Cryptography, Hankerson et al., Springer 2004. A summary of such operations and properties appears in paragraphs [0012] through [0026] of U.S. Pat. No. 7,483,533, issued Jan. 27, 2009, which are hereby incorporated by reference.

Elliptic curve cryptography is gaining increasing acceptance in public-key applications, such as public key exchange, digital signatures, etc. because elliptic curve cryptography permits smaller key size than RSA and other such algorithms, and the difficulty of attacking or solving the EC discrete logarithm problem. However, elliptic curve cryptography is not widely used for symmetric encryption systems, block ciphers, or encryption of long messages.

The difficulty with such applications arises from the process for encrypting a plaintext message using an elliptic curve. In conventional elliptic curve cryptography, conventional bit string embedding into an elliptic curve point involves and iterative algorithm to search for an x-coordinate value that will lead to a quadratic residue value of the y-coordinate starting from an initial x-coordinate value specified by the bit string to be embedded. The process normally requires several iterations, and the number of iterations needed is different for different bit strings that are being embedded. This process incurs considerable overhead in time and in memory storage, since look-up tables are often used.

Also, in conventional elliptic curve cryptography, an elliptic curve polynomial defined over a finite field that needs N-bits for representation of its elements, only ((n+nx+1)N-L) bits
of the message data bits can be embedded in any elliptic polynomial point, where \( I \) is the number of extra bits that do not carry data.

U.S. Patent No. 7,483,533, issued Jan. 27, 2009 to Ibrahim (one of the present inventors), which is hereby incorporated by reference in its entirety, describes a cryptographic system based upon Ibrahim’s elliptic polynomial equation, which has more than one \( x \)-coordinate into which data may be embedded. The ’533 patent describes methods for using an elliptic polynomial with more than one \( x \)-coordinate for digital signatures, public-key cryptography, and at least one method for use in symmetric or shared key cryptography. However, there still exists a need for an efficient, secure method for encrypting messages for symmetric encryption systems, and one that is particularly suitable for block ciphers.

Thus, a method of performing elliptic polynomial cryptography with elliptic polynomial hopping solving the aforementioned problems is desired.

SUMMARY OF THE INVENTION

The method of performing elliptic polynomial cryptography with elliptic polynomial hopping contemplates improving the security of existing block ciphers by using elliptic polynomial cryptography. Different elliptic polynomials are used to encrypt different blocks of the same plaintext. The method provides for communicating securely over an insecure channel using symmetric key elliptic polynomial cryptography, which uses an elliptic polynomial with more than one independent \( x \)-coordinate, as described in U.S. Patent No. 7,483,533, issued Jan. 27, 2009 to Ibrahim, the contents of which are incorporated by reference in their entirety.

More specifically, given a set of elliptic polynomial points that satisfy an elliptic polynomial equation with more than one independent \( x \)-coordinate defined over a finite field \( F \) of the following form:

one of the variables, termed the \( y \)-coordinate, has a maximum degree of two, and appears on its own in only one of the monomials;
the other variables, termed the \( x \)-coordinates, have a maximum degree of three, and each must appear in at least one of the monomials with a degree of three; and all monomials that contain \( x \)-coordinates must have a total degree of three;
and given that the group of points of the elliptic polynomial with the above form are defined over additions in the extended dimensional space described in the ’533 patent, the new method contemplates the use of elliptic polynomial hopping, wherein different elliptic polynomials are used to encrypt different blocks of the same plaintext, and wherein the number of \( x \)-coordinates and the underlying finite field used could also be varied and selected from a pre-defined set.

By using elliptic polynomial cryptography with more than one \( x \)-coordinate, the additional \( x \)-coordinates may be used to embed extra message data bits in a single elliptic point that satisfies an elliptic polynomial equation. Given that \( nx \) additional \( x \)-coordinates are used, with \( nx \) greater or equal to one, a resulting elliptic point has \((nx+1)\) \( x \)-coordinates, wherein all coordinates are elements of a finite field \( F \). The number of points that satisfy an elliptic polynomial equation with \( nx \) additional \( x \)-coordinates defined over \( F \) and that can be used in the corresponding cryptosystem is increased by a factor of \((#F)^{(n-1)}\), where \( #F \) denotes the size of \( F \).

This provides increasing security, wherein different elliptic polynomial are used to encrypt successive message blocks, and wherein each elliptic polynomial used to encrypt each message block is selected at random using an initial secret key and a random number generator. The only information known to an attacker would be the size of the ciphertext blocks.

The elliptic polynomial hopping method disclosed herein is made possible, wherein it is shown that the given form of the elliptic polynomial equation described above ensures that an elliptic polynomial and its twist are isomorphic to each other.

The embedding method allows the embedding of a bit string into the \( x \)-coordinates of an elliptic polynomial point in a deterministic and non-iterative manner when the elliptic polynomial has the above-described form. This new deterministic and non-iterative embedding method requires less time overhead than conventional iterative embedding methods used in existing elliptic curve polynomial cryptography, and also does not require a different number of iterations for different bit strings that are being embedded. This makes the method suitable for block ciphers.

The isomorphic relationship between an elliptic polynomial and its twist, which is obtained as a result of the given form of the elliptic polynomial equation, ensures that any bit string whose equivalent binary value is an element of the underlying finite field has a bijective relationship between the bit string and a point, which is either on an elliptic polynomial or its twist. This bijective relationship allows the development of elliptic polynomial hopping-based encryption.

Furthermore, up to \((nx+ny+1)\) \( N \) bits (i.e. more than \((nx+ny+1)\) \( N \) \( L \) bits) can be encrypted in each elliptic polynomial point. In the classical approach of elliptic curve polynomial cryptography encryption and decryption, the security of the resulting cryptosystem relies on breaking the elliptic polynomial discrete logarithm problem, which can be summarized as: given the points \( k(x_0, y_0; x_1, y_1; \ldots; x_n, y_n) \) and \( (x_0, y_0; x_1, y_1; \ldots; x_n, y_n) \), find the scalar \( k \).

In the present method, different elliptic polynomials are used to encrypt successive message blocks, and the elliptic polynomials used to encrypt each succeeding message block are selected at random using an initial secret key and a random number generator. Since the elliptic polynomials used to encrypt each successive message block is not known, this makes the elliptic polynomial discrete logarithm problem much more difficult to solve, since the attacker does not know the elliptic polynomial coefficients that are needed to compute point additions and point doublings. The harder problem of the solving the discrete logarithm problem when the used elliptic polynomial is not known can be used either to increase security for the same finite field, or to reduce complexity for the same security by using a finite field with a smaller order.

In addition, all of the elliptic polynomial cryptography-based cipher block chaining methods disclosed herein are scalable, i.e., the key size may be changed without changing the underlying algorithm.

These and other features of the present invention will become readily apparent upon further review of the following specification and drawings.

BRIEF DESCRIPTION OF THE DRAWINGS

The sole drawing FIGURE is a block diagram illustrating system components for implementing the method of performing elliptic polynomial cryptography with elliptic polynomial hopping according to the present invention.

These and other features of the present invention will become readily apparent upon further review of the following specification and drawings.
DETAILED DESCRIPTION OF THE PREFERRED EMBODIMENTS

The method of performing elliptic polynomial cryptography with elliptic polynomial hopping provides elliptic polynomial cryptographic methods based on the elliptic polynomial discrete logarithm problem. It is well known that an elliptic polynomial discrete logarithm problem is a computationally "difficult" or "hard" problem.

The elliptic polynomial encryption methods to be described below use elliptic polynomials in their generation, where different elliptic polynomials are used for different blocks of the same plaintext. Particularly, the cryptographic methods use an elliptic polynomial with more than one independent x-coordinate. More specifically, a set of elliptic polynomial points are used which satisfy an elliptic polynomial equation with more than one independent x-coordinate which is defined over a finite field $F$ having the following properties:

One of the variables (the y-coordinate) has a maximum degree of two, and appears on its own in only one of the monomials; the other variables (the x-coordinates) have a maximum degree of three, and each must appear in at least one of the monomials with a degree of three; and all monomials which contain x-coordinates must have a total degree of three.

The group of points of the elliptic polynomial with the above form is defined over additions in the extended dimensional space, and, as will be described in detail below, the method makes use of elliptic polynomials where different elliptic polynomials are used for different blocks of the same plaintext.

The particular advantage of using elliptic polynomial cryptography with more than one x-coordinate is that additional x-coordinates are used to embed extra message data bits in a single elliptic point that satisfies the elliptic polynomials equation. Given that nx additional x-coordinates are used, with nx being greater than or equal to one, a resulting elliptic point has $(nx+1)$ x-coordinates, where all coordinates are elements of the finite field $F$. The number of points which satisfy an elliptic polynomials equation with nx additional x-coordinates defined over $F$ and which can be used in the corresponding cryptosystem is increased by a factor of $(#)^m$, where # denotes the size of a field.

Through the use of this particular method, security is increased through the usage of different elliptic polynomials for different message blocks during the encryption. Further, the elliptic polynomial used for each message block is selected at random, preferably using an initial value and a random number generator.

Given the form of the elliptic polynomial equation described above, the elliptic polynomial and its twist are isomorphic with respect to one another. The method uses an embedding technique to be described in greater detail below, which allows the embedding of a bit string into the x-coordinates of an elliptic polynomial point in a deterministic and non-iterative manner when the elliptic polynomial has the above described form. This embedding method overcomes the disadvantage of the time overhead of the iterative embedding methods used in existing elliptic polynomial-based methods.

The difficulty of using conventional elliptic polynomial cryptography for message encryption typically lies in the iterative and non-deterministic method needed to embed a bit string into an elliptic polynomial point, which has the drawback of the number of iterations needed being different for different bit strings which are being embedded. As a consequence, different calculation times are required for different blocks of bit strings. Such a data-dependent generation time is not suitable for generalized encryption methods, which require data independent encryption times. Further, with regard to iterative and non-deterministic methods in conventional elliptic polynomial cryptography, given an elliptic polynomial defined over a finite field that needs N-bits for the representation of its elements, only $(nx+ny+1)N+1$ bits of the message data bits can be embedded in any elliptic polynomial point.

The isomorphic relationship between an elliptic polynomial and its twist, which is obtained as a result of the given form of the elliptic polynomial equation, ensures that any bit string whose equivalent binary value is an element of the underlying finite field has a bijective relationship between the bit string and a point which is either on the elliptic polynomial or its twist. This bijective relationship allows for the development of the encryption to be described below.

In the conventional approach to elliptic polynomial cryptography, the security of the resulting cryptosystem relies on breaking the elliptic polynomial discrete logarithm problem, which can be summarized as: given the points $(x_0,x_1,\ldots,x_{nx+1},y_0)$ and $(x_0',x_1',\ldots,x_{nx+1}',y_0')$, find the scalar k.

Further, projective coordinates are used at the sending and receiving entities in order to eliminate inversion or division during each point addition and doubling operation of the scalar multiplication. It should be noted that all of present methods disclosed herein are scalable.

In the following, with regard to elliptic polynomials, the “degree” of a variable u is simply the exponent i. A polynomial is defined as the sum of several terms, which are herein referred to as “monomials”, and the total degree of a monomial $u^iv^jw^k$ is given by $(i+j+k)$. Further, in the following, the symbol F denotes set membership.

One form of the subject elliptic polynomial equation with more than one x-coordinate and one or more y-coordinates is defined as follows: the elliptic polynomial is a polynomial with more than two independent variables such that the maximum total degree of any monomial in the curve is three; at least two or more of the variables, termed the x-coordinates, have a maximum degree of three, and each must appear in at least one of the monomials with a degree of three; and at least one or more variables, termed the y-coordinates, have a maximum degree of two, and each must appear in at least one of the monomials with a degree of two.

Letting $S_{nx}$ represents the set of numbers from 0 to nx (i.e., $S_{nx}=[0,\ldots,nx]$), and letting $S_{ny}$ represents the set of numbers from 0 to ny (i.e., $S_{ny}=[0,\ldots,ny]$), and further setting $(nx+ny+1)$, then, given a finite field, F, the following equation defined over F is one example of the polynomial described above:

$$\sum_{i=1}^{nx} a_i x_i^2 + \sum_{i=1}^{nx} a_{1i} y_i x_i + \sum_{i=1}^{nx} a_{2i} y_i^2 + \sum_{i=1}^{nx} a_{3i} y_i^3 + \sum_{i,j=1}^{nx} c_{ij} y_i y_j x_i + (1),$$

where

$$a_{1i}, a_{2i}, a_{3i}, c_{ij}, x_i, y_i \in F.$$
With regard to the use of the elliptic polynomial equation in the addition of points of an elliptic polynomial with more than one x-coordinate and one or more y-coordinates, we may examine specific coefficients $a_{12}, a_{22}, b_{12}, a_{13}, b_{13}$, and $b_{23}$, for $F$, wherein a set of points $EC^{(m+xn+1)}, y_n$ is defined as the $(m+xn+1)$-uple $(x_0, x_1, \ldots, x_m, x_{m+1}, x_{m+2})$, wherein $x_i \in F$, $i = 0, 1, 2$, and $k \in \mathbb{K}_F$. This set of points is solutions of $F$, though excluding the point $O(0, 0, \ldots, 0)$ and the point at infinity, $(x_0, x_1, \ldots, x_m, y_0, y_1, \ldots, y_m)$. The rules for conventional elliptic curve point addition may be adopted to define an additive binary operation $+$, over $EC^{(m+xn+1)}, y_n$, i.e., for all

$$(x_0, x_1, \ldots, x_m, y_0, y_1, \ldots, y_m) \in EC^{(m+xn+1)}, y_n$$

and

$$(x_0, x_1, \ldots, x_m, y_0, y_1, \ldots, y_m) \in EC^{(m+xn+1)}, y_n$$

the sum:

$$(x_0, x_1, \ldots, x_m, y_0, y_1, \ldots, y_m) = (x_0, x_1, \ldots, x_m, y_0, y_1, \ldots, y_m) + (x_0, x_1, \ldots, x_m, y_0, y_1, \ldots, y_m)$$

if the following axioms:

(i) There exists a set $(x_0, x_1, \ldots, x_m, y_0, y_1, \ldots, y_m) \in EC^{(m+xn+1)}, y_n$ such that $(x_0, x_1, \ldots, x_m, y_0, y_1, \ldots, y_m) + (x_0, x_1, \ldots, x_m, y_0, y_1, \ldots, y_m) = (x_0, x_1, \ldots, x_m, y_0, y_1, \ldots, y_m)$ for all $(x_0, x_1, \ldots, x_m, y_0, y_1, \ldots, y_m) \in EC^{(m+xn+1)}, y_n$.

(ii) For every set $(x_0, x_1, \ldots, x_m, y_0, y_1, \ldots, y_m) \in EC^{(m+xn+1)}, y_n$ there exists an inverse set, $-(x_0, x_1, \ldots, x_m, y_0, y_1, \ldots, y_m) \in EC^{(m+xn+1)}, y_n$, such that $(x_0, x_1, \ldots, x_m, y_0, y_1, \ldots, y_m) + (x_0, x_1, \ldots, x_m, y_0, y_1, \ldots, y_m) = (x_0, x_1, \ldots, x_m, y_0, y_1, \ldots, y_m)$.

(iii) The additive binary operation in $EC^{(m+xn+1)}, y_n$ is commutative, and the p-group $PC^{(m+xn+1), y_n}$ forms a group over addition when

(iv) the additive binary operation in $EC^{(m+xn+1)}, y_n$ is associative.

Prior to a more detailed analysis of the above axioms, the concept of point equivalence must be further developed. Mappings can be used to indicate that an elliptic point represented using $(mx+1)$ x-coordinates and $(ny+1)$ y-coordinates, $(x_0, x_1, \ldots, x_m, y_0, y_1, \ldots, y_m)$, is equivalent to one or more elliptic points that satisfy the same elliptic polynomial equation, including the equivalence of an elliptic point to itself.

Points that are equivalent to one another can be substituted for each other at random, or according to certain rules during point addition and/or point doubling operations. For example, the addition of two points $(x_0, x_1, \ldots, x_m, y_0, y_1, \ldots, y_m)$ and $(x_0, x_1, \ldots, x_m, y_0, y_1, \ldots, y_m)$ is given by:

$$(x_0, x_1, \ldots, x_m, y_0, y_1, \ldots, y_m) + (x_0, x_1, \ldots, x_m, y_0, y_1, \ldots, y_m) = (x_0, x_1, \ldots, x_m, y_0, y_1, \ldots, y_m)$$

If the point $(x_0, x_1, \ldots, x_m, y_0, y_1, \ldots, y_m)$ is equivalent to the point $(x_0, x_1, \ldots, x_m, y_0, y_1, \ldots, y_m)$, then the former can be substituted for $(x_0, x_1, \ldots, x_m, y_0, y_1, \ldots, y_m)$ in the above equation in order to obtain:

$$(x_0, x_1, \ldots, x_m, y_0, y_1, \ldots, y_m) + (x_0, x_1, \ldots, x_m, y_0, y_1, \ldots, y_m) = (x_0, x_1, \ldots, x_m, y_0, y_1, \ldots, y_m)$$

Mappings that are used to define equivalences can be based on certain properties that exist in elliptic polynomial equations, such as symmetry between variables. As an example, we can consider the point $(x_0, x_1, y_0, y_1)$ that satisfies the equation $y_0 = x_0^2 + x_1 y_0 + x_2$. The equivalent of this point may be defined as $(x_1, x_0, y_0, y_1)$.

With regard to the addition rules for $EC^{(m+xn+1), y_n}$, the addition operation of two points $(x_0, x_1, \ldots, x_m, y_0, y_1, y_2, \ldots, y_n)$ and $(x_0, x_1, \ldots, x_m, y_0, y_1, y_2, \ldots, y_n)$, otherwise expressed as:

$$(x_0, x_1, \ldots, x_m, y_0, y_1, y_2, \ldots, y_n) + (x_0, x_1, \ldots, x_m, y_0, y_1, y_2, \ldots, y_n) = (x_0, x_1, \ldots, x_m, y_0, y_1, y_2, \ldots, y_n)$$

is calculated in the following manner. First, a straight line is drawn which passes through the two points to be added. The straight line intersects $EC^{(m+xn+1), y_n}$ at a third point, which we denote as $(x_0, x_1, \ldots, x_m, y_0, y_1, y_2, \ldots, y_n)$. The sum point is defined as $(x_0, x_1, \ldots, x_m, y_0, y_1, y_2, \ldots, y_n)$. From the above definition of the addition rule, addition over $EC^{(m+xn+1), y_n}$ is commutative, that is:

$$(x_0, x_1, \ldots, x_m, y_0, y_1, y_2, \ldots, y_n) + (x_0, x_1, \ldots, x_m, y_0, y_1, y_2, \ldots, y_n) = (x_0, x_1, \ldots, x_m, y_0, y_1, y_2, \ldots, y_n) + (x_0, x_1, \ldots, x_m, y_0, y_1, y_2, \ldots, y_n)$$

for all $(x_0, x_1, \ldots, x_m, y_0, y_1, y_2, \ldots, y_n), (x_0, x_1, \ldots, x_m, y_0, y_1, y_2, \ldots, y_n), (x_0, x_1, \ldots, x_m, y_0, y_1, y_2, \ldots, y_n), (x_0, x_1, \ldots, x_m, y_0, y_1, y_2, \ldots, y_n) \in EC^{(m+xn+1), y_n}$. This commutativity satisfies axiom (iii) above.

There are two primary cases that need to be considered for the computation of point addition for $EC^{(m+xn+1), y_n}$: (A) for at least one $i \in \mathbb{N}_x$, $x_i = x_i$, and (B) for all $i \in \mathbb{N}_x$, $x_i = \infty$. Case B includes three sub-cases:

i. For all $i \in \mathbb{N}_x, x_i = x_i$, which corresponds to point doubling.

ii. For all $i \in \mathbb{N}_x, x_i = x_i$, and $y_0, y_2, y_4, \ldots$ are the roots of the following quadratic equation in $y_2'$:

$$(a_0 y_0^2 + \sum_{i=k}^{n} a_i y_i^2 + y_0) \left( \sum_{i=k}^{n} a_i y_i^2 + y_0 + \sum_{i=k}^{n} a_i y_i^2 + y_0 \right) + \sum_{i=k}^{n} a_i y_i^2 + y_0$$

iii. For all $i \in \mathbb{N}_x, x_i = \infty$, and $y_0, y_2, y_4, \ldots$ are the roots of the following quadratic equation in $y_2'$:

$$y_0 \left( \sum_{i=k}^{n} c_i y_i^2 + y_0 \right) \left( \sum_{i=k}^{n} c_i y_i^2 + y_0 \right) + \sum_{i=k}^{n} c_i y_i^2 + y_0$$
\[
\frac{y_k - y_{k,1}}{y_{k,2} - y_{k,1}} = \frac{x_j - x_{j,1}}{x_{j,2} - x_{j,1}}
\]

for \(k \in S_{xy}\) and \(j \in S_{xy}\), and

\[
\frac{x_j - x_{j,1}}{x_{j,2} - x_{j,1}} = \frac{x_j - x_{j,1}}{x_{j,2} - x_{j,1}}
\]

for \(i \in S_{xy}\).

For this case, \(y_k = m_k x_j + c_{yk}\), where

\[
m_{yk} = \frac{y_{k,2} - y_{k,1}}{x_{j,2} - x_{j,1}}
\]

and \(c_{yk} = y_{k,1} - m_{yk} x_{j,1}\). Further, \(x_j = m_{xj} x_j + c_{xj}\), where

\[
m_{xj} = \frac{x_{j,2} - x_{j,1}}{x_{j,2} - x_{j,1}}
\]

and \(c_{xj} = x_{j,1} - m_{xj} x_{j,1}\). Equation (1) can then be re-written as:

\[
\sum_{k \in S_{xy}} a_{k1} y_k^2 + \sum_{k \in S_{xy}} a_{k2} y_k + \sum_{k \in S_{xy}} a_{k3} y_k + x_j \sum_{k \in S_{xy}} c_{yk} y_k + x_j \sum_{k \in S_{xy}} c_{yk} y_k + \sum_{k \in S_{xy}} c_{yk} y_k + x_j \sum_{k \in S_{xy}} c_{yk} y_k + \sum_{k \in S_{xy}} c_{yk} y_k + \sum_{k \in S_{xy}} c_{yk} y_k = 0
\]

Expanding the terms in the above equation leads to a cubic equation in \(x_j\): \(C_3 x_j^3 + C_2 x_j^2 + C_1 x_j + C_0 = 0\) where \(C_3, C_2, C_1, \) and \(C_0\) are obtained from the above equation.

Assuming \(C_3 \neq 0\), the above cubic equation in \(x_j\) has three roots \(x_{j,1}, x_{j,2}, \) and \(x_{j,3}\), and can be written as \((x_j - x_{j,1})(x_j - x_{j,2})(x_j - x_{j,3}) = 0\). Normalizing by the coefficient of \(x_j^3\) and equating the coefficients of \(x_j^3\) in the resulting equation with that in \((x_j - x_{j,1})(x_j - x_{j,2})(x_j - x_{j,3}) = 0\), one obtains a solution for \(x_{j,3}\):

\[
x_{j,3} = \frac{-C_2}{C_3} - x_{j,1} - x_{j,2}
\]

The values of \(y_{k,3}, k \in S_{xy}\), and \(x_{j,3}, i \in S_{xy}\) may be similarly obtained from equations for \(x_j - x_{j,3}\).

For cases where \(C_3 = 0\), \(C_2 x_j^2 + C_1 x_j + C_0 = 0\) becomes a quadratic equation. Such quadratic equations may be used in the definition of point equivalences.

With regard to Case B for all \(j \in S_{xy}\), the three sub-cases are considered below. In all cases, \(y_{k,3} = x_{j,3} - x_{j,2} \in S_{xy}\).

For Case B.i.i., all \(k \in S_{xy}\), \(y_{k,3} = x_{j,3} - x_{j,2}\), which corresponds to point doubling. In this case, \((x_{j,3}, x_{j,3}, \ldots, x_{j,3}, y_{k,3}, \ldots, y_{k,3}) \in S_{xy}\).

The sum is written as

\[
\sum_{k \in S_{xy}} a_{k1} (x_k x_j + c_{yk})^2 + \sum_{k \in S_{xy}} a_{k2} (x_k x_j + c_{yk}) (x_k x_j + c_{yk}) + \sum_{k \in S_{xy}} a_{k3} (x_k x_j + c_{yk}) (x_k x_j + c_{yk}) + \sum_{k \in S_{xy}} a_{k4} (x_k x_j + c_{yk}) (x_k x_j + c_{yk}) + \sum_{k \in S_{xy}} a_{k5} (x_k x_j + c_{yk}) (x_k x_j + c_{yk}) = 0
\]

There are several ways of defining the addition in this case. Three possible rules are described below. Case B.i.i.: Letting \(S_{xy} \subseteq S_{xy}\), let \(S_{xy} \subseteq S_{xy}\), and \(0 \in S_{xy}\), computing the value of \(x_j\) and \(y_{k,3}\) as at least one, then the straight line in this case can be defined as a tangent to the
point \((x_{0,0}, y_{1,0}, \ldots, x_{m,n}, y_{l,m})\) defined in a sub-dimensional space with coordinates \(y_u\) and \(x_u\), with \(n \in \mathbb{S}_{m,n}^{l,m}\) and \(m \in \mathbb{S}_{m,n}^{l,m}\).

In this case, the gradients \(m_y\) and \(m_x\) of the straight line to be used in equation (2) are essentially the first derivatives of \(y_u\) and \(x_u\), respectively, \(n \in \mathbb{S}_{m,n}^{l,m}\) and \(m \in \mathbb{S}_{m,n}^{l,m}\), for \(F\) with respect to \(x_u, y_u\), i.e.,

\[
m_y = \frac{dy_u}{dx_u} \text{ and } m_x = \frac{dx_u}{dy_u}.
\]

Using these derivatives for the values of the gradients,

\[
m_y = \frac{dy_u}{dx_u},
\]

where \(n \in \mathbb{S}_{m,n}^{l,m}\) and \(m \in \mathbb{S}_{m,n}^{l,m}\), in equation (2) and noting that it is assumed that

\[
dx_u = 0,
\]

for \(m \in \mathbb{S}_{m,n}^{l,m}\) and

\[
dy_u = 0,
\]

for \(n \in \mathbb{S}_{m,n}^{l,m}\), then a solution for \(x_u, y_u\) may be obtained.

The values of \(y_u\) for \(n \in \mathbb{S}_{m,n}\), and \(x_u\) for \(m \in \mathbb{S}_{m,n}\), \& \(m \in \mathbb{S}_{m,n}\), can further be obtained for \(y_x, y_{x'}\). The choice of the \(x_u\)-coordinates, \(m \in \mathbb{S}_{m,n}^{l,m}\), and \(y_u\)-coordinates, \(n \in \mathbb{S}_{m,n}^{l,m}\), which can be used to compute the tangent of the straight line in this case may be chosen at random or according to a pre-defined rule. Further, a different choice of the \(x_u\)-coordinates, \(m \in \mathbb{S}_{m,n}^{l,m}\), and \(y_u\)-coordinates, \(n \in \mathbb{S}_{m,n}^{l,m}\), may be made when one needs to compute successive point doublings, such as that needed in scalar multiplication.

With regard to the next case, Case B.1.2, the second possible way of defining the addition of a point with itself is to apply a sequence of the point doublings according to the rule defined above in Case B.1.1, where the rule is applied with a different selection of the \(x_u\)-coordinate(s) and \(y_u\)-coordinate(s) in each step of this sequence.

In the third sub-case, Case B.1.3, a point is substituted with one of its equivalents. Letting \((x_{0,0}, y_{1,0}, \ldots, x_{m,n}, y_{l,m})\) represent the equivalent point of \((x_{0,0}, y_{1,0}, \ldots, x_{m,n}, y_{l,m})\), then equation (3) may be written as:

\[
(x_{0,0}, y_{1,0}, \ldots, x_{m,n}, y_{l,m}) = (x_{0,0}, y_{1,0}, \ldots, x_{m,n}, y_{l,m}).
\]

With regard to Case B.2, for \(k \in \mathbb{S}_{m,n}\) and \(k \neq 0\), \(y_{k,1} = -y_{k,2}\), and where \(y_{0,1}\) & \(y_{0,2}\) are the roots of the quadratic equation in \(y_{0,1}\), this case corresponds to generation of the point inverse.

Letting \(y_{k,1} = -y_{k,2}\) for \(k \in \mathbb{S}_{m,n}\) and \(k \neq 0\), then any two points, such as the point \((x_{0,0}, y_{1,0}), \ldots, x_{m,n}, y_{l,m})\) and \((y_{0,1}, y_{0,2})\) in \(\mathbb{E}_{m,n}^{l,m}\) and the point \((x_{0,0}, y_{1,0}, \ldots, x_{m,n}, y_{l,m})\) and \((y_{0,1}, y_{0,2})\), \(k \in \mathbb{S}_{m,n}\) and \(k \neq 0\), these two points lying on the hyperplane of \(x_{0,0}, y_{1,0}, \ldots, x_{m,n}, y_{l,m})\) and \((y_{0,1}, y_{0,2})\), \(k \in \mathbb{S}_{m,n}\) and \(k \neq 0\), this straight line joining these two points such that \((x_{0,0}, y_{1,0}, \ldots, x_{m,n}, y_{l,m})\) \((y_{0,1}, y_{0,2})\), \(k \in \mathbb{S}_{m,n}\) and \(k \neq 0\), is also in this hyperplane.

Substituting the values of \(x_{0,0}, y_{1,0}, \ldots, x_{m,n}, y_{l,m})\) and \((y_{0,1}, y_{0,2})\), in an elliptic polynomial equation with multiple \(x\)-coordinates and multiple \(y\)-coordinates, a quadratic equation for \(y_{0,1}\) is obtained, as given above. Thus, \(y_{0,1}\) has only two solutions, \(y_{0,1} = y_{0,2}\).

Thus, a line joining points \((x_{0,0}, y_{1,0}, \ldots, x_{m,n}, y_{l,m})\) \((y_{0,1}, y_{0,2})\) \(E_{m,n}^{l,m}\) and \((x_{0,0}, y_{1,0}, \ldots, x_{m,n}, y_{l,m})\) \((y_{0,1}, y_{0,2})\) \(E_{m,n}^{l,m}\), does not intersect with \(E_{m,n}^{l,m}\) at a third point.

A line that joins these two points is assumed to intersect with \(E_{m,n}^{l,m}\) at infinity \((x_{0,0}, y_{1,0}, \ldots, x_{m,n}, y_{l,m})\) \((y_{0,1}, y_{0,2})\), \(E_{m,n}^{l,m}\). This point at infinity is used to define both the inverse of a point in \(E_{m,n}^{l,m}\) and the identity point. According to the addition rule defined above, one can write:

\[
(x_{0,0}, y_{1,0}, \ldots, x_{m,n}, y_{l,m}) = (y_{0,1}, y_{0,2}) = (x_{0,0}, y_{1,0}, \ldots, x_{m,n}, y_{l,m}), \tag{4}
\]

since the third point of intersection of such lines is assumed to be at infinity, \((x_{0,0}, y_{1,0}, \ldots, x_{m,n}, y_{l,m}) \in E_{m,n}^{l,m}\). Thus, this equation defines a unique inverse for any point \((x_{0,0}, y_{1,0}, \ldots, x_{m,n}, y_{l,m}) \in E_{m,n}^{l,m}\), namely:

\[
-(x_{0,0}, y_{1,0}, \ldots, x_{m,n}, y_{l,m}) = (y_{0,1}, y_{0,2}). \tag{5}
\]

Thus, equation (4) can be written as:

\[
(x_{0,0}, y_{1,0}, \ldots, x_{m,n}, y_{l,m}) = (y_{0,1}, y_{0,2}) = (x_{0,0}, y_{1,0}, \ldots, x_{m,n}, y_{l,m}), \tag{5}
\]

Further, a line joining the point at infinity \((x_{0,0}, y_{1,0}, \ldots, x_{m,n}, y_{l,m}) \in E_{m,n}^{l,m}\) and a point \((x_{0,0}, y_{1,0}, \ldots, x_{m,n}, y_{l,m}) \in E_{m,n}^{l,m}\) will intersect with \(E_{m,n}^{l,m}\) at \((x_{0,0}, y_{1,0}, \ldots, x_{m,n}, y_{l,m}) \in E_{m,n}^{l,m}\). Thus, from the addition rule defined above,

\[
(x_{0,0}, y_{1,0}, \ldots, x_{m,n}, y_{l,m}) = (y_{0,1}, y_{0,2}), \tag{6}
\]

Equation (5) satisfies axiom (ii) while equation (6) satisfies axiom (i), defined above.

Case B.iii applies for all other conditions except those in cases B.i and B.ii. This case occurs only when \(y_{0,1}\) is greater than or equal to one. Given two points \((x_{0,0}, y_{1,0}, \ldots, x_{m,n}, y_{l,m}) \in E_{m,n}^{l,m}\) and \((x_{0,0}, y_{1,0}, \ldots, x_{m,n}, y_{l,m}) \in E_{m,n}^{l,m}\) that do not satisfy the conditions of cases B.i and B.ii above, the sum point is written as \((x_{0,0}, y_{1,0}, \ldots, x_{m,n}, y_{l,m}) \in E_{m,n}^{l,m}\). There are several possible rules to find the sum point in this case. Three possible methods are given below.

1) Using three point doublings and one point addition,

\[
(x_{0,0}, y_{1,0}, \ldots, x_{m,n}, y_{l,m}) \in E_{m,n}^{l,m}. \tag{5}
\]

2) Using one point doublings and three point additions,

\[
(x_{0,0}, y_{1,0}, \ldots, x_{m,n}, y_{l,m}) \in E_{m,n}^{l,m}. \tag{6}
\]
3) using point equivalence,

\[
\begin{align*}
(x_{0,1}, y_{0,1}, \ldots, x_{m,1}, y_{m,1}, \ldots, y_{n,1}, x_{0,2}, y_{0,2}, \ldots, x_{m,2}, y_{m,2}, \ldots, y_{n,2}) &= (x_{0,1}, y_{0,1}, \ldots, x_{m,1}, y_{m,1}, \ldots, y_{n,1}) \\
x_{i,1} &= x_{i,2}, \quad y_{i,1} = y_{i,2}, \quad \text{for} \quad i \in \mathbb{Z}_{m, n}
\end{align*}
\]

where \((x_{0,1}, y_{0,1}, \ldots, x_{m,1}, y_{m,1}, \ldots, y_{n,1}, x_{0,2}, y_{0,2}, \ldots, x_{m,2}, y_{m,2}, \ldots, y_{n,2})\) is assumed to be the equivalent point of \((x_{0,1}, y_{0,1}, \ldots, x_{m,1}, y_{m,1}, \ldots, y_{n,1}, x_{0,2}, y_{0,2}, \ldots, x_{m,2}, y_{m,2}, \ldots, y_{n,2})\).

It should be noted that the above methods for defining the sum point are not the only ones that can be defined and are provided for exemplary purposes only. The choice of method used to obtain the sum point in this case should depend on the computation complexity of point addition and point doubling.

With regard to associativity, one way of proving associativity of \((E_{\text{mixed}})^{\pm 1} \cdot 2\) is as follows: Given particular elliptic polynomial equations (i.e., for particular coefficients \(a,b,c,d,e,f\), \(g,h,k,l,m,n\), \(p,q,r,s\), \(t,u,v,w,x,y,z\)) defined over a finite field \(F\), it can be shown that any point \(Q \in E_{\text{mixed}}^{\pm 1} \cdot 2\) (and any of its equivalent points) can be uniquely written as \(kQ \cdot P \in E_{\text{mixed}}^{\pm 1} \cdot 2\), where \(P\) is the generator point of \((E_{\text{mixed}})^{\pm 1} \cdot 2\), then the corresponding \(Q\) groups based on such polynomials are associative.

This is because any three points \(Q,R,S \in E_{\text{mixed}}^{\pm 1} \cdot 2\) can be written as \(kQ \cdot P \in E_{\text{mixed}}^{\pm 1} \cdot 2\), respectively, thus their sum \((Q+R+S) = (kQ+P+kP+Q) = (kQ+P+R+S) = 0\) can be carried out in any order.

The following elliptic polynomial equation with \(n=1\) and \(my=0\) is used to show an example of the equations in Case A used in point addition: \(y_{x}^2 = x_{x}^3 + ax_{x} + c\). Choosing \(y_{x} = -x_{x}\), and substituting \(y_{x} = mx_{x} + c\) from Case A above for \(y_{x}\), and the corresponding equation \(y_{x} = mx_{x} + c\), one obtains \(y_{x}^2 = x_{x}^3 + (mx_{x} + c)^2 + mx_{x}\).\n
Expanding this equation yields the equation \(m_{x}^2x_{x}^2 + 2m_{x}c_{x}x_{x} + c_{x}^2 = x_{x}^3 + (mx_{x} + c)^2 + mx_{x}\).\n
From equation (2), the solution for \(x_{0,3}\) in this case is obtained:

\[
x_{0,3} = \left(\frac{2m_{x}c_{x}x_{x} + c_{x}^2}{1 + m_{x}^2}\right) - x_{x} - x_{x}.
\]

Similarly, one can obtain the values of \(y_{0,3}\) and \(x_{1,3}\) for \(x_{0,3} = x_{0,3}\).

It should be noted that when \(m_{x} = -1\), the coefficient of the cubic term in the above is zero; i.e., \(C_{x} = 0\). In this case, the resulting quadratic equation can be used in the definition of point equivalences for the points that satisfy the elliptic polynomial equation.

Each of the equations for point addition and point doublings derived for cases A and B above require modular inversion or division. In cases where field inversions or divisions are significantly more expensive (in terms of computational time and energy) than multiplication, projective coordinates are used to remove the requirement for field inversion or division from these equations.

Several projective coordinates can be utilized. In the preferred embodiment, the Jacobean projective coordinate system is used. As an example, we examine:

\[
x_{i} = \frac{x_{i}}{v_{i}} \quad \text{for} \quad i \in \mathbb{Z}_{m,n}; \quad \text{and} \quad \text{and}
\]

\[
\begin{align*}
\left(\frac{x_{0,1}}{v_{0,1}}, \ldots, \frac{x_{m,1}}{v_{m,1}}, \frac{y_{0,1}}{v_{0,1}}, \ldots, \frac{y_{n,1}}{v_{n,1}}\right) &= (8) \\
\text{Using Jacobian projection yields:} \\
\sum_{k \in \mathbb{Z}_{m,n}} a_{0,k} \frac{y_{i}^2}{v_{i}^2} + \sum_{k \in \mathbb{Z}_{m,n}} a_{2,k} \frac{y_{i}^2}{v_{i}^2} + \sum_{k \in \mathbb{Z}_{m,n}} a_{4,k} \frac{y_{i}^2}{v_{i}^2} + \\
\sum_{k \in \mathbb{Z}_{m,n}} c_{0,k} \frac{y_{i}^2}{v_{i}^2} + \sum_{k \in \mathbb{Z}_{m,n}} c_{2,k} \frac{y_{i}^2}{v_{i}^2} + \sum_{k \in \mathbb{Z}_{m,n}} c_{4,k} \frac{y_{i}^2}{v_{i}^2} + \\
\sum_{k \in \mathbb{Z}_{m,n}} b_{0,k} \frac{x_{i}^2}{v_{i}^2} + \sum_{k \in \mathbb{Z}_{m,n}} b_{2,k} \frac{x_{i}^2}{v_{i}^2} + \sum_{k \in \mathbb{Z}_{m,n}} b_{4,k} \frac{x_{i}^2}{v_{i}^2} + b_{i}.
\end{align*}
\]

which can be rewritten as:

\[
\begin{align*}
\sum_{k \in \mathbb{Z}_{m,n}} a_{0,k} \frac{y_{i}^2}{v_{i}^2} + \sum_{k \in \mathbb{Z}_{m,n}} a_{2,k} \frac{y_{i}^2}{v_{i}^2} + \sum_{k \in \mathbb{Z}_{m,n}} a_{4,k} \frac{y_{i}^2}{v_{i}^2} + \\
\sum_{k \in \mathbb{Z}_{m,n}} c_{0,k} \frac{y_{i}^2}{v_{i}^2} + \sum_{k \in \mathbb{Z}_{m,n}} c_{2,k} \frac{y_{i}^2}{v_{i}^2} + \sum_{k \in \mathbb{Z}_{m,n}} c_{4,k} \frac{y_{i}^2}{v_{i}^2} + \\
\sum_{k \in \mathbb{Z}_{m,n}} b_{0,k} \frac{x_{i}^2}{v_{i}^2} + \sum_{k \in \mathbb{Z}_{m,n}} b_{2,k} \frac{x_{i}^2}{v_{i}^2} + \sum_{k \in \mathbb{Z}_{m,n}} b_{4,k} \frac{x_{i}^2}{v_{i}^2} + b_{i}.
\end{align*}
\]

In the following, the points \((x_{0,1}, x_{1,1}, \ldots, x_{m,1}, y_{0,1}, \ldots, y_{n,1}, v_{0,1}, v_{1,1}, \ldots, v_{m,1})\) are assumed to satisfy equation (10). When \(v = 0\), the projected point \((x_{0,1}, x_{1,1}, \ldots, x_{m,1}, y_{0,1}, \ldots, y_{n,1}, v_{0,1}, v_{1,1}, \ldots, v_{m,1})\) corresponds to the point:

\[
\left(\frac{x_{0}}{v_{0}}, \ldots, x_{m}, y_{0}, \ldots, y_{n}\right) = \left(\frac{x_{0}}{v_{0}}, \frac{x_{1}}{v_{1}}, \ldots, x_{m}, \frac{y_{0}}{v_{0}}, \frac{y_{1}}{v_{1}}, \ldots, \frac{y_{n}}{v_{n}}\right)
\]

which satisfies equation (1).

Using Jacobean projective coordinates, equation (10) can be written as:

\[
\left(\frac{x_{0,1}}{v_{0,1}}, \ldots, \frac{x_{m,1}}{v_{m,1}}, \frac{y_{0,1}}{v_{0,1}}, \frac{y_{1,1}}{v_{1,1}}, \ldots, \frac{y_{n,1}}{v_{n,1}}\right) = (11)
\]

\[
\left(\frac{x_{0,1}}{v_{0,1}}, \ldots, \frac{x_{m,1}}{v_{m,1}}, \frac{y_{0,1}}{v_{0,1}}, \frac{y_{1,1}}{v_{1,1}}, \ldots, \frac{y_{n,1}}{v_{n,1}}\right) = \left(\frac{x_{0,2}}{v_{0,2}}, \ldots, \frac{x_{m,2}}{v_{m,2}}, \frac{y_{0,2}}{v_{0,2}}, \frac{y_{1,2}}{v_{1,2}}, \ldots, \frac{y_{n,2}}{v_{n,2}}\right)
\]

By using Jacobean projective coordinates in the equations of Cases A and B above, and by an appropriate choice of the value of \(v_{i}\), it can be shown that point doubling and point addition can be computed without the need for field inversion or division.
As described above, conventional bit string embedding into an elliptic curve point involves an iterative algorithm to search for an x-coordinate value which will lead to a quadratic residue value of the y-coordinate starting from an initial x-coordinate value specified by the bit string to be embedded. However, such a process requires that the number of iterations needed is different for different bit strings that are being embedded. In the present method, an embedding methodology is utilized that embeds a bit string into an appropriate elliptic polynomial point with (nx+1) x-coordinates and (ny+1) y-coordinates in a deterministic and non-iterative manner. Further, the elliptic polynomial equation is of a specified form, i.e., it is isomorphic to its twist. This method circumvents the need for an iterative algorithm that involves the usual search for a quadratic residue value of the y-coordinate (which typically requires several iterations) and, further, suffers from the drawback that the number of iterations needed is different for different bit strings that are being embedded.

In order to examine the embedding method, the twist of an elliptic polynomial equation needs to be defined. Given an elliptic polynomial with (nx+1) x-coordinates and (ny+1) y-coordinates of the form described above:

\[ y^2 + \sum_{i < nx} a_i x^i + \sum_{i < ny} a_{ny-i} x^i = \sum_{i < nx} b_i x^i + \sum_{i < ny} b_{ny-i} x^i, \]

where \(a_1, a_2, b_1, b_2 \in F\).

Given certain values for the x-coordinates \(x_{1,1}, \ldots, x_{nx,1}\) and y-coordinates \(y_{1,1}, \ldots, y_{ny,1}\), respectively, that are elements of the finite field, these values are substituted into the elliptic polynomial equation (1) in order to obtain a quadratic equation in \(y_i\):

\[ y_i^2 = -\sum_{i < nx} a_i x_{i,1}^{2i} - \sum_{i < ny} a_{ny-i} x_{i,1}^{2i} + \sum_{i < nx} b_i x_{i,1}^{2i} + \sum_{i < ny} b_{ny-i} x_{i,1}^{2i} = T. \]

If a solution of the above quadratic equation (i.e., \(y_i^2 = T\)) is an element of the finite field \(F\), the point \((x_{i,1}, y_{1,1}, \ldots, x_{nx,1}, y_{ny,1})\) is said to satisfy the given elliptic polynomial equation. If a solution of the above quadratic equation is not an element of the finite field \(F\), the point \((x_{i,1}, y_{1,1}, \ldots, x_{nx,1}, y_{ny,1})\) is said to satisfy the twist of the given elliptic curve equation. The inventive embedding method is based on the isomorphic relationship between a curve and its twist as described in the following theorem:

An elliptic polynomial equation of the form given above is isomorphic to its twist if:

1) there are mathematical mappings that can be defined on the values of \(x_{1,1}, \ldots, x_{nx,1}, y_{1,1}, \ldots, y_{ny}\) (i.e., \(\phi_x(x_i)\) and \(\phi_y(y_i)\)) such that any point \((x_{1,1}, \ldots, x_{nx,1}, y_{1,1}, \ldots, y_{ny})\) that satisfies such an elliptic polynomial equation can be mapped into another point \((\phi_x(x_{i,1}), \phi_y(y_{1,1}), \ldots, \phi_x(x_{nx,1}), \phi_y(y_{ny,1}))\) that satisfies the twist of the same elliptic polynomial equation; and
2) the mapping between the points \((x_{1,1}, \ldots, x_{nx,1}, y_{1,1}, \ldots, y_{ny})\) and \((\phi_x(x_{1,1}), \phi_y(y_{1,1}), \ldots, \phi_x(x_{nx,1}), \phi_y(y_{ny,1}))\) is unique, i.e., a one-to-one correspondence.

The proof of this theorem is as follows. Re-writing equation (12) as:

\[ y_i^2 = \sum_{i < nx} a_i x_{i,1}^{2i} + \sum_{i < ny} a_{ny-i} x_{i,1}^{2i} + \sum_{i < nx} b_i x_{i,1}^{2i} + \sum_{i < ny} b_{ny-i} x_{i,1}^{2i}, \]

and letting the right-hand side of equation (13) be denoted as \(T\), then:

\[ T = \sum_{i < nx} a_i x_{i,1}^{2i} + \sum_{i < ny} a_{ny-i} x_{i,1}^{2i} + \sum_{i < nx} b_i x_{i,1}^{2i} + \sum_{i < ny} b_{ny-i} x_{i,1}^{2i}, \]

Thus, any value of \(x_{1,1}, \ldots, x_{nx,1}, y_{1,1}, \ldots, y_{ny}\) will lead to a value of \(\in F\). \(T\) could be quadratic residue or non-quadratic residue. If \(T\) is quadratic residue, then equation (14) is written as:

\[ T_q = \sum_{i < nx} a_i x_{i,1}^{2i} + \sum_{i < ny} a_{ny-i} x_{i,1}^{2i} + \sum_{i < nx} b_i x_{i,1}^{2i} + \sum_{i < ny} b_{ny-i} x_{i,1}^{2i}, \]

where \(x_{1,1}, \ldots, x_{nx,1}, y_{1,1}, \ldots, y_{ny}\in F\) denotes the values of \(x_{1,1}, \ldots, x_{nx,1}, y_{1,1}, \ldots, y_{ny}\) that result in a quadratic residue value of \(T\), which is hereafter denoted as \(T_q\).

If \(T\) is non-quadratic residue, then equation (14) is written as:

\[ T_q = \sum_{i < nx} a_i x_{i,1}^{2i} + \sum_{i < ny} a_{ny-i} x_{i,1}^{2i} + \sum_{i < nx} b_i x_{i,1}^{2i} + \sum_{i < ny} b_{ny-i} x_{i,1}^{2i}, \]

where \(x_{1,1}, \ldots, x_{nx,1}, y_{1,1}, \ldots, y_{ny}\in F\) denotes the values of \(x_{1,1}, \ldots, x_{nx,1}, y_{1,1}, \ldots, y_{ny}\) that result in a non-quadratic residue value of \(T\), denoted as \(T_q\).

Letting \(g\) be any non-quadratic residue number in \(F\) (i.e., \(g \in F\) & \(g \not\in F\)), then multiplying equation (15) with \(g^3\) yields:

\[ T_q g^3 = -g^3 \sum_{i < nx} a_i x_{i,1}^{2i} - g^3 \sum_{i < ny} a_{ny-i} x_{i,1}^{2i} + g^3 \sum_{i < nx} b_i x_{i,1}^{2i} + g^3 \sum_{i < ny} b_{ny-i} x_{i,1}^{2i}, \]

which can be re-written as:

\[ T_q g^3 = g^3 \sum_{i < nx} a_i x_{i,1}^{2i} + g^3 \sum_{i < ny} a_{ny-i} x_{i,1}^{2i} + g^3 \sum_{i < nx} b_i x_{i,1}^{2i} + g^3 \sum_{i < ny} b_{ny-i} x_{i,1}^{2i}, \]

6) the mapping between the points \((x_{1,1}, \ldots, x_{nx,1}, y_{1,1}, \ldots, y_{ny})\) and \((\phi_x(x_{1,1}), \phi_y(y_{1,1}), \ldots, \phi_x(x_{nx,1}), \phi_y(y_{ny,1}))\) is unique, i.e., a one-to-one correspondence.
It should be noted that if \( g \) is non-quadratic residue, then \( g^2 \) is also non-quadratic residue. Further, the result of multiplying a quadratic residue number by a non-quadratic residue number is a non-quadratic residue number. Thus, \( g^2 + T_y \) is non-quadratic residue.

By comparing the terms of equations (16) and (17), we obtain the following mappings:

\[ x_i \rightarrow g^x_i \]  
(18); and

\[ y_i \rightarrow k^y_i \]  
(19); and

\[ T_i \rightarrow T_y \]  
(20).

The mappings between the variables and \( x_i \) and \( y_i \) in equation (18), \( x_i \) and \( y_i \) in equation (19), and \( T_i \) and \( T_y \) in equation (20) are all bijective, i.e., there is a one-to-one correspondence from basic finite field arithmetic. As a consequence, the mappings between the \((n+x+y+2)\)-tuple \( (x_i, x_i', \ldots, x_{i+k}, y_{i+y}, \ldots, y_{i+y+n}) \) and the \((n+x+y+2)\)-tuple \( (y_i, y_i', \ldots, y_{i+y}, T_y, y_{i+y}, \ldots, y_{i+y+n}) \) are also bijective.

Therefore, for every solution of equation (15), there is an isomorphic solution that satisfies equation (16), and since the mappings of the coordinates of one to the other are given in equations (18)-(20), these two solutions are isomorphic with respect to each other.

Since \( T_y \) is quadratic residue, this expression can be written as:

\[ T_i \rightarrow T_y^2 \]  
(21)

Thus, from equation (20), \( T_y \) can be written as:

\[ T_y \rightarrow T_y^2 \]  
(22).

Using equations (21) and (22), equations (15) and (16) can be written as:

\[ y_i^2 = \sum_{k=k_0} c_i x_{i+k} + \sum_{k=k_0} c_{i+k} x_{i+k} \]  
(23)

and

\[ g^2 y_i^2 = \sum_{k=k_0} c_i x_{i+k} + \sum_{k=k_0} c_{i+k} x_{i+k} \]  
(24).

Since any solution of equation (15) has an isomorphic solution that satisfies equation (16), it follows that the solution of equation (23), denoted as \( (x_i, x_i', \ldots, x_{i+k}, y_{i+y}, \ldots, y_{i+y+n}) \), has an isomorphic solution that satisfies equation (24), denoted \( (g(x_i), g(x_i'), \ldots, g(x_{i+k}), g^2 y_{i+y}, \ldots, g^2 y_{i+y+n}) \).

The solutions of equation (23) form the points \( (x_i, x_i', \ldots, x_{i+k}, y_{i+y}, \ldots, y_{i+y+n}) \) that satisfy an elliptic polynomial. Similarly, the solutions of equation (24) form the points \( (g(x_i), g(x_i'), \ldots, g(x_{i+k}), g^2 y_{i+y}, \ldots, g^2 y_{i+y+n}) \) that satisfy its twist. This proves the above theorem.

An example of a mapping of the solutions of equation (23) defined over \( F(p) \), where \( p=3 \mod 4 \), to the solutions of its twist is implemented by using \(-x_i\) for the \( x \)-coordinates and \(-y_i^2\) for the \( y \)-coordinates.

The isomorphism between an elliptic polynomial and its twist, discussed above, is exploited for the embedding of the bit string of a shared secret key into the appropriate \( x \) and \( y \) coordinates of an elliptic polynomial without the need for an iterative search for a quadratic residue value of the corresponding \( y_o \)-coordinate, which usually requires several iterations, where the number of iterations needed is different for different bit strings which are being embedded.

Assuming \( F=F(p) \) and that the secret key is a \( m \)-bit string such that \((n+x+y+1)N>M=N-1\), where \( N \) is the number of bits needed to represent the elements of \( F(p) \), then the secret key bit string is divided into \((n+x+y+1)\) bit-strings \( k_{x,0}, k_{x,1}, \ldots, k_{x,y}, k_{y,0}, k_{y,1}, \ldots, k_{y,y} \). The value of the bit-strings \( k_{x,0}, k_{x,1}, \ldots, k_{x,y}, k_{y,0}, k_{y,1}, \ldots, k_{y,y} \) must be less than \( p \). In the preferred embodiment of embedding the \((n+x+y+1)\) bit-strings \( k_{x,0}, k_{x,1}, \ldots, k_{x,y}, k_{y,0}, k_{y,1}, \ldots, k_{y,y} \) the embedding is as follows:

First, assign the value of the bit string of \( k_{x,0}, k_{x,1}, \ldots, k_{x,y} \) to \( x_{i,0}, x_{i,1}, \ldots, x_{i,y} \). Next, assign the value of the bit string of \( k_{y,0}, k_{y,1}, \ldots, k_{y,y} \) to \( y_{i,0}, y_{i,1}, \ldots, y_{i,y} \). Then compute:

\[ T_i = x_{i,0} y_{i,0} + \sum_{k=k_0} c_i x_{i+k} + \sum_{k=k_0} c_{i+k} x_{i+k} \]  
(25)

Finally, use the Legendre test to see if \( T \) has a square root. If \( T \) has a square root, assign one of the roots to \( y_o \); otherwise, use the \( x \)-coordinates and \( y \)-coordinates of the elliptic polynomial point with the embedded shared secret key bit string are given by \( x_{i,0} \) and \( g^{2y}_{i,0} \), respectively, where \( g \) is non-quadratic residue in \( F \).

It should be noted that \( p \) is usually predetermined prior to encryption, so that the value of \( g \) can also be predetermined. Further, the receiver can identify whether the point \( (x_{i,0}, x_{i,1}, \ldots, y_{i,0}, y_{i,1}, \ldots, y_{i,0}+y_{i,1}) = (x_{i,0}, x_{i,1}, \ldots, y_{i,0}, y_{i,1}) \) or the point \( (g(x_{i,0}), g(x_{i,1}), \ldots, g(y_{i,0}), g(y_{i,1}), \ldots, g^2 y_{i,0}, g^2 y_{i,1}) \) is the elliptic polynomial point with the embedded secret key bit strings without any additional information. Additionally, any non-quadratic value in \( F(p) \) can be used for \( g \). For efficiency, \( g \) is chosen to be \(-1\) for \( p=3 \mod 4 \) and \( g \) is chosen to be \( 2 \) for \( p=1 \mod 4 \).

The same deterministic and non-iterative method described above can be used to embed a secret message bit string into an elliptic polynomial point in a deterministic and non-iterative manner. Assuming \( F=F(p) \) and that the message is an \( M \)-bit string such that \((n+x+y+1)N>M=N-1\), where \( N \) is the number of bits needed to represent the elements of \( F(p) \), then the message bit string is divided into \((n+x+y+1)\) bit-strings \( m_{i,0}, m_{i,1}, \ldots, m_{i,y}, m_{i,0}, m_{i,1}, \ldots, m_{i,y} \). The value of the bit-strings \( m_{i,0}, m_{i,1}, \ldots, m_{i,y}, m_{i,0}, m_{i,1}, \ldots, m_{i,y} \) must be less than \( p \). As in the previous embodiment, the embedding of the \((n+x+y+1)\) bit-strings \( m_{i,0}, m_{i,1}, \ldots, m_{i,y}, m_{i,0}, m_{i,1}, \ldots, m_{i,y} \) can be accomplished as follows:

First, assign the value of the bit string of \( m_{i,0}, m_{i,1}, \ldots, m_{i,y} \) to \( x_{i,0}, x_{i,1}, \ldots, x_{i,y} \). Next, assign the value of the bit string of \( m_{i,0}, m_{i,1}, \ldots, m_{i,y} \) to \( y_{i,0}, y_{i,1}, \ldots, y_{i,y} \). Then compute:

\[ T_i = x_{i,0} y_{i,0} + \sum_{k=k_0} c_i x_{i+k} + \sum_{k=k_0} c_{i+k} x_{i+k} \]  
(26)

Finally, use the Legendre test to see if \( T \) has a square root. If \( T \) has a square root, then assign one of the roots to \( y_o \) other-
Elliptic polynomial hopping refers to hopping the point $(x_{1,o}, x_{2,o}, \ldots, x_{n,o}, 0, \ldots, 0)$ that satisfies the one elliptic polynomial (for example, the $s$-th elliptic polynomial with coefficients $b_{1}^{(s)}, b_{2}^{(s)} \in F$) into an equivalent point $(x_{1,r}, x_{2,r}, \ldots, x_{n,r}, 0, \ldots, 0)$ that satisfies the $r$-th elliptic polynomial with coefficients $b_{1}^{(r)}, b_{2}^{(r)} \in F$.

One method of achieving this is as follows. First, set the $x$-coordinates of the hopped point $x_{1,o}, x_{2,o}, \ldots, x_{n,o}$ to the $x$-coordinates $x_{1,r}, x_{2,r}, \ldots, x_{n,r}$ of the original point, $x_{1,o} \rightarrow x_{1,r}$ for $i=0, \ldots, n$. Next, substitute the value of the $x$-coordinates $x_{1,o}, x_{2,o}, \ldots, x_{n,o}$ into the new elliptic polynomial equation to obtain

$$T^{(n)} = \sum_{i=0}^{n} [b_{1}^{(i)}(x_{1}^{(i)}) + \sum_{i=0}^{n} b_{2}^{(i)}(x_{1}^{(i)})^2 x_{1}^{(i)}]$$

(any value of $x_{1,r}, x_{2,r}, \ldots, x_{n,r}$ will lead to a value of $T^{(n)}\in F(p)$, where $T^{(n)}$ could be quadratic residue or non-quadratic residue). Finally, if $T^{(n)}$ is quadratic residue, set $y_{0}^{(n)} = \sqrt{T^{(n)}}$ and the hopped point is given by $(y_{0}^{(n)}, x_{1,r}, x_{2,r}, \ldots, x_{n,r}, 0, \ldots, 0)$; otherwise, if $T^{(n)}$ is a non-quadratic residue, set $y_{0}^{(n)} = \sqrt{-1} \sqrt{T^{(n)}}$ and the hopped point is given by $(y_{0}^{(n)}, x_{1,r}, x_{2,r}, \ldots, x_{n,r}, 0, \ldots, 0)$.

Thus, any point that satisfies an elliptic polynomial can be hopped to an equivalent point on another elliptic polynomial, even if the two polynomials are not isomorphic to each other.

Further, a point is never mapped to another point that satisfies the twist of another elliptic polynomial. As can be seen in the final step above, a point that satisfies an elliptic polynomial is mapped (hopped) to another point that satisfies another elliptic polynomial. Any point that satisfies one elliptic polynomial can be uniquely mapped to another point that satisfies either the equation of an elliptic polynomial or the equation of its twist. In order to show this unique mapping, an additional “tuple” must be used to indicate as to whether a point that satisfies an elliptic polynomial is mapped to a point on another elliptic polynomial or the twist of this other elliptic polynomial.

Thus, for purposes of point mapping between one elliptic polynomial into another, a point is represented as $(x_{1,o}, x_{2,o}, \ldots, x_{n,o}, y_{o}, \alpha_{o})$, and the last variable, $\alpha_{o}$, indicates whether the point $(x_{1,o}, x_{2,o}, \ldots, x_{n,o}, y_{o}, \alpha_{o})$ satisfies the previous elliptic polynomial was mapped to an elliptic or its twist. If $\alpha_{o} = 1$, the point $(x_{1,o}, x_{2,o}, \ldots, x_{n,o}, y_{o}, \alpha_{o})$ was originally mapped to a point on the elliptic polynomial, otherwise if $\alpha_{o} = -1$, the point $(x_{1,o}, x_{2,o}, \ldots, x_{n,o}, y_{o}, \alpha_{o})$ was mapped to a point on the twist of an elliptic polynomial. The addition of the variable $\alpha_{o}$ as an extra “tuple” in the representation of points allows the above procedure to be reversed as follows.

First, if $\alpha_{o} = 1$, the $x$-coordinates $x_{1,o}, x_{2,o}, \ldots, x_{n,o}$ are given by $x_{1,o} = x_{1,r}$ for $i=0, \ldots, n$; otherwise, if $\alpha_{o} = -1$, the $x$-coordinates $x_{1,o}, x_{2,o}, \ldots, x_{n,o}$ are given by $x_{1,o} = x_{1,r}$ for $i=0, \ldots, n$. Next, substitute the value of the $x$-coordinates $x_{1,o}, x_{2,o}, \ldots, x_{n,o}$ into the new $(s-1)$ elliptic polynomial equation to obtain

$$T^{(s-1)} = \sum_{i=0}^{s-1} [b_{1}^{(i)}(x_{1}^{(i)}) + \sum_{i=0}^{s-1} b_{2}^{(i)}(x_{1}^{(i)})^2 x_{1}^{(i)}]$$

where the $s$-th elliptic polynomial is not necessarily isomorphic to the $s$-th elliptic polynomial, i.e., where all or some of the coefficients $b_{1}^{(s)} b_{2}^{(s)} \in F$ are different and independent of the coefficients $b_{1}^{(r)} b_{2}^{(r)} \in F$. 

$$T^{(n)} = \sum_{i=0}^{n} [b_{1}^{(i)}(x_{1}^{(i)}) + \sum_{i=0}^{n} b_{2}^{(i)}(x_{1}^{(i)})^2 x_{1}^{(i)}]$$
Finally, compute $y^{(t+1)} = y^{(t+1)}$, since it is known that the original point $(x_0^{(t+1)}, y_0^{(t+1)})$, \ldots, $(x_n^{(t+1)}, y_n^{(t+1)})$ must satisfy the equation of an elliptic polynomial. Thus, $T^{(t+1)}$ is always a quadratic residue.

It should be noted that in the above procedures, the value of $\alpha^{(t+1)}$ is not defined, since the value depends on the mapping of the point that satisfies the $(s+2)$-elliptic polynomial into the $(s+1)$-elliptic polynomial. This value of $\alpha^{(t+1)}$ must be provided as additional information.

The preferred embodiment of the present method includes the following steps:

a) determining a maximum block size that can be embedded into $(m+1)$-x-coordinates and n-y-coordinates, wherein $m$ is an integer, and setting the maximum block size to be $(m+n+1)$ in bits, wherein $n$ is an integer;

b) a sending correspondent and a receiving correspondent agree upon the values of $m$ and $n$, and further agree on a set of coefficients $b_0, b_1, \ldots, b_{n+y}$, wherein $F$ represents a finite field where the field’s elements can be represented in $K$-bits, the sending and receiving correspondents further agreeing upon a random number $k_0$, which is at least part of a shared secret key for communication, a random number generator, a random number $k_0$, that is a portion of the shared secret key used for communication, and a set of $(m+1)$ numbers such that $x_0, x_1, \ldots, x_n$, the set being used to find an initial base point, the set being made public; the sending correspondent then performs the following steps:

c) generating at least a portion of the coefficients $b_0, b_1, \ldots, b_n$ of an initial elliptic polynomial to be used for the message authentication code of a 0-th message block from the shared secret key $k_0$;

d) embedding the set of $(m+1)$ numbers $x_0, x_1, \ldots, x_n$ into an elliptic polynomial point to obtain an initial base point $(x_0, y_0), (x_1, y_1), \ldots, (x_n, y_n)$ using a data embedding method;

e) computing the 0-th block of the message bit string into an elliptic polynomial message point $(x_0, y_0), (x_1, y_1), \ldots, (x_n, y_n)$ using the data embedding method;

f) computing a scalar multiplication of the 0-th block shared key $k_0$ with the base point $(x_0, y_0), (x_1, y_1), \ldots, (x_n, y_n)$ as $k_0, k_1, \ldots, k_n$ using a data embedding method;

g) computing a cipher point of the 0-th block data $(x_0, y_0), (x_1, y_1), \ldots, (x_n, y_n)$ as $k_0, k_1, \ldots, k_n$ using a data embedding method;

h) establishing integers $j$ and $u$, such that $j=1, \ldots, u$, and initializing the integer $j$ as $j=1$ and repeating the following steps i) to m), and incrementing $j$ at each step until all of the message data blocks are processed:

i) generating a random number $k_0$ with the random number generator based upon $k_0$;

j) generating at least a portion of the coefficients $b_0, b_1, \ldots, b_{n+y}$ of the j-th elliptic polynomial from the random number $k_0$;

k) embedding the j-th block of the message bit string into a j-th elliptic polynomial message point $(x_0, y_0), (x_1, y_1), \ldots, (x_n, y_n)$ using the data embedding method;

l) hopping the scalar multiplication point $(x_0, y_0), (x_1, y_1), \ldots, (x_n, y_n)$ to an equivalent message authentication code point $(x_0, y_0), (x_1, y_1), \ldots, (x_n, y_n)$ that satisfies the j-th elliptic polynomial selected in step u) using the data embedding method;

m) computing the message point of the j-th received data block $(x_0, y_0), (x_1, y_1), \ldots, (x_n, y_n)$ as $k_0, k_1, \ldots, k_n$ using a data embedding method;

n) initializing the integer $j$ as $j=1$ and repeating the following steps i) to m), and incrementing $j$ at each step until all of the message data blocks are processed.
In an alternative embodiment of the encryption method, the underlying finite field, the number of x-coordinates and the monomials used are selected from a predefined set of elliptic polynomial equations that are agreed upon between the corresponding entities.

The primary condition is that the ciphertext size must be the same as the block size. The ciphertext block size is determined by the maximum block size defined by the selected elliptic polynomial equations. If certain elliptic equations result in ciphertext block sizes that are smaller than the specified size of the ciphertext, then padding is used to maintain uniform block size. The alternative method includes the steps of:

a) defining a maximum block size that can be embedded into (nx+1) x-coordinates and ny y-coordinates, wherein n is an integer, and setting the maximum block size to be (nx+ny+1)N bits, wherein N is an integer;

b) a sending correspondent and a receiving correspondent agree upon the values of nx and ny, and further agree on a set of coefficients a, b∈Ε, wherein F represents a finite field where the field’s elements can be represented in N-bits, the sending and receiving correspondents further agree on a random number k₀, which is at least part of the shared secret key for communication, a random number generator, a random number k₀, that is a portion of the shared secret key used for communication, and a set of (nx+1) numbers such that xₙ∈Ε, wherein i=nx, the set being used to find an initial base point, the set being made public;

the sending correspondent then performs the following steps:

c) selecting an elliptic polynomial form from the agreed upon set of general form equations using the shared secret key of the initial block, which is denoted as the 0-th block, and generating at least a portion of a set of coefficients b₁, b₂∈Ε of the first elliptic polynomial to be used for a message authentication code of the 0-th message block from the shared secret key k₀;

d) embedding the set of (nx+1) numbers xₙ∈Ε into an elliptic polynomial point to obtain an initial base point (x₀, x₁, . . . , xₙ, xₙ₋₁, . . . , xₙ₋₄, yₙ₋₃, yₙ₋₃/yₙ₋₄, α₀, α₀) using a data embedding method;

e) embedding the 0-th block of the message bit string into an elliptic polynomial message point (x₀, x₁, . . . , yₙ₋₃, yₙ₋₃/yₙ₋₄, α₀, α₀) using the data embedding method;

f) computing a scalar multiplication of the 0-th block shared key k₀ with the base point (x₀, x₁, . . . , yₙ₋₃, yₙ₋₃/yₙ₋₄, α₀, α₀) as:

\[
\kappa_0 x_{0, 1} \ldots \kappa_0 y_{0, 3} \ldots \kappa_0 \alpha_0
\]

g) computing a cipher point of the 0-th data block (x₀, x₁, . . . , yₙ₋₃, yₙ₋₃/yₙ₋₄, α₀, α₀) as:

\[
(x₀, x₁, \ldots, xₙ₋₃, yₙ₋₃/yₙ₋₄, \alpha₀, \alpha₀)
\]

wherein α₀ = α₀ and sending a set of appropriate bits of the x-coordinates and a bit indicating the value of α₀ of the cipher point (x₀, x₁, . . . , yₙ₋₃, yₙ₋₃/yₙ₋₄, α₀, α₀) to the receiving correspondent;

h) establishing integers i and u, such that j=1, . . . , u, and initializing the integer j = 1 and repeating the following steps (i) to (m), and incrementing j at each step until all of the message data blocks are processed:

i) generating a random number kp, with the random number generator based upon kp; i, and selecting an elliptic polynomial form from the agreed upon set of general form equations using the shared secret key of the j-th block;

j) generating at least a portion of the coefficients b₁, b₂∈Ε of the j-th elliptic polynomial from the random number kp;

k) embedding the j-th block of the message bit string into a j-th elliptic polynomial message point (x₀, x₁, . . . , yₙ₋₃, yₙ₋₃/yₙ₋₄, α₀, α₀) using the data embedding method;

l) hopping the scalar multiplication point (x₀, x₁, . . . , yₙ₋₃, yₙ₋₃/yₙ₋₄, α₀, α₀) by (j−1, j−1, . . . , yₙ₋₃/yₙ₋₄, α₀, α₀) to an equivalent message authentication code point (x₀, x₁, . . . , yₙ₋₃, yₙ₋₃/yₙ₋₄, α₀, α₀) that satisfies the j-th elliptic polynomial selected in step j) using the data embedding method;

m) computing the cipher point of the j-th data block (x₀, x₁, . . . , yₙ₋₃, yₙ₋₃/yₙ₋₄, α₀, α₀) as:

\[
(x₀, x₁, \ldots, xₙ₋₃, yₙ₋₃/yₙ₋₄, \alpha₀, \alpha₀)
\]

n) appropriate bits of the x-coordinates and a bit indicating the value of α₀ of the cipher point (x₀, x₁, . . . , yₙ₋₃, yₙ₋₃/yₙ₋₄, α₀, α₀) being sent to the receiving correspondent; the receiving correspondent then performs the following steps:

o) selecting an elliptic polynomial form from the agreed upon set of general form equations using the shared secret key of the initial block and generating at least a portion of the coefficients b₁, b₂∈Ε of an elliptic polynomial to be used for the message authentication code of the 0-th message block from the shared secret key kp;

p) embedding the set of (nx+1) numbers xₙ∈Ε into an elliptic polynomial point to obtain an initial base point (x₀, x₁, . . . , yₙ₋₃, yₙ₋₃/yₙ₋₄, α₀, α₀) using the data embedding method;

q) computing the scalar multiplication of the 0-th block shared key k₀ with the base point (x₀, x₁, . . . , yₙ₋₃, yₙ₋₃/yₙ₋₄, α₀, α₀) as:

\[
(x₀, x₁, \ldots, xₙ₋₃, yₙ₋₃/yₙ₋₄, \alpha₀, \alpha₀)
\]

r) computing the message point of the 0-th data block (x₀, x₁, . . . , yₙ₋₃, yₙ₋₃/yₙ₋₄, α₀, α₀) as:

\[
(x₀, x₁, \ldots, xₙ₋₃, yₙ₋₃/yₙ₋₄, \alpha₀, \alpha₀)
\]

wherein α₀ = α₀ and recovering the secret message bit string for the 0-th block from appropriate x-coordinates of the message point (x₀, x₁, . . . , yₙ₋₃, yₙ₋₃/yₙ₋₄, α₀, α₀) if α₀ = 1, and recovering the secret message bit string from a point (gxₙ₋₃, gyₙ₋₃, . . . , gₙ₋₃, yₙ₋₃/yₙ₋₄) if α₀ if ε = ε;

s) initializing the integer j as 1 and repeating the following steps (t) to (x), and incrementing each step until all of the message data blocks are processed;

t) generating a random number kp, with the random number generator based upon kp; i, and selecting an elliptic polynomial form from the agreed upon set of general form equations using the shared secret key of the j-th block;

u) generating at least a portion of the coefficients b₁, b₂∈Ε of the j-th elliptic polynomial from the random number kp, and selecting an elliptic polynomial form from the agreed upon set of general form equations using the shared secret key of the j-th block;

v) hopping the scalar multiplication point (x₀, x₁, . . . , yₙ₋₃, yₙ₋₃/yₙ₋₄, α₀, α₀) by (j−1, j−1, . . . , yₙ₋₃/yₙ₋₄, α₀, α₀) to an equivalent message authentication code point (x₀, x₁, . . . , yₙ₋₃, yₙ₋₃/yₙ₋₄, α₀, α₀) that satisfies the j-th elliptic polynomial selected in step j) using the data embedding method;

w) computing the message point of the j-th received data block (x₀, x₁, . . . , yₙ₋₃, yₙ₋₃/yₙ₋₄, α₀, α₀) is computed as:
wherein $\alpha_m^{(g)} = \alpha_m^{(0)}$; and

x) recovering the secret message bit string for the j-th block from appropriate x-coordinates of the message point $(x_0,y_0)$,

$\alpha_1^{(0)}$, $\ldots$, $\alpha_m^{(0)}$, $\gamma^{(0)}$, $\alpha_m^{(0)}$, if $\alpha_m^{(0)} = 1$, and recovering the secret message bit string from the point $(gx_0,y_0)$,

$\gamma^{(0)}$, $\ldots$, $\gamma^{(0)}$, $\gamma^{(0)}$, $\gamma^{(0)}$, $\gamma^{(0)}$, if $\alpha_m^{(g)} = 1$.

As noted above, the methods include data embedding. In order to embed a message bit string into a point $(x, \sqrt{cy})$ which satisfies either an elliptic curve equation $y^2 = x^3 + ax + b$ or its twist, $cy^2 = x^3 + ax + b$, the message bit string is first divided into N-bit strings and the $m$th block is denoted as $m$. Following this, the value of the bit string of $m$ is assigned to $x_m$, and the values of $x_m$ are substituted and the value of $t_m$ is computed using $t_m = x_m^{2} + ax_m + b$.

If $t_m$ is quadratic residue, then $t_m$ is the point and the point is given as $(x_m, y_m)$. However, if $t_m$ is non-quadratic residue, then

$$y_m = \sqrt{t_m}$$

and the point is given as $(x_m, y_m)$. The message point is then denoted as $(x_m, y_m)$, where the point is on the elliptic curve if $\alpha_m = 1$, and the point is on the twist if $\alpha_m = 1$.

Alternatively, data embedding may be performed as follows:

a) dividing the message bit string into $(m + m' + 1)$ bitstrings $m_0, m_1, \ldots, m_n, m_{n+1}, \ldots, m_{n+m'}$;

b) assigning the value of the bit string of $m_0, m_1, \ldots, m_n$ to $x_0, x_1, \ldots, x_n$;

c) assigning the value of the bit string of $m_{n+1}, \ldots, m_{n+m'}$ to $y_1, \ldots, y_{n+m'}$;

d) computing

$$T^{(m') + 1} = \sum_{i=0}^{n} b_{i1} x_i^2 + \sum_{i=0}^{n} b_{i2} y_i^2 x_i$$

and

e) performing a Legendre test to determine if $T$ has a square root, wherein if $T$ has a square root, then assigning the square root to $y_0$, and if $T$ does not have a square root, then the x-coordinates and y-coordinates of the elliptic curve point with the embedded shared secret key bit string are selected as $gx_{n+m}$ and $g^x_{n+m}$, respectively, where g is non-quadratic residue in $F$.

As a hybrid of these techniques, the embedding may be performed as:

a) dividing the message bit string into $N$-bit strings, wherein the $i$-th block is given by $m$;

b) assigning the value of the bit string of $m_i$ to $x_m$;

c) substituting the values of $x_m$ and computing a residue value $t_m$ as $t_m = x_m^2 + ax_m + b$, wherein if $t_m$ is quadratic residue such that $y_m = \sqrt{t_m}$, then the message point is embedded as $(x_m, y_m)$, and if $t_m$ is non-quadratic residue such that

$$y_m = \sqrt{t_m}$$

then the message point is embedded $(x_m, y_m)$.

The Legendre Symbol is used to test whether an element of $p$ has a square root or not, i.e., whether an element is quadratic residue or not. The Legendre Symbol and test are as follows. Given an element of a finite field $F(p)$, such as $d$, the Legendre symbol is defined as

$$\left( \frac{d}{p} \right)$$

In order to test whether $d$ is a quadratic residue or not, the Legendre symbol,

$$\left( \frac{d}{p} \right)$$

is computed such that

$$\left( \frac{d}{p} \right) = \begin{cases} +1 & \text{if } x \text{ is quadratic residue} \\ 0 & \text{if } x \equiv 0 \pmod{p} \\ -1 & \text{otherwise} \end{cases}$$

In the above, the password protocols use the scalar multiplication $K_{m}(x_0,y_0)$. It should be noted that, in order to find a collision means, that there are two message bit strings $m$ and $m'$ such that their integer values $k_m$ and $k_m'$ will lead to $k_m(x_0,y_0) = k_m(x_0,y_0)$. This collision implies that integers can be found such that $k_m - k_m' = \#E$, where $\#E$ is the order of the group $(E,C)$. This is equivalent to solving the elliptic polynomial discrete logarithm problem. This also applies to finding a collision for the points on the twist of an elliptic curve, $k_m(x_0,y_0) = k_m(x_0,y_0)$. Thus, the security of the password protocols depends on the security of the underlying elliptic polynomial cryptography.

The security of elliptic curve cryptosystems is assessed by both the effect on the solution of the elliptic polynomial discrete logarithmic problem (ECDCP) and power analysis attacks.

It is well known that the elliptic polynomial discrete logarithm problem (ECDCP) is apparently intractable for non-singular elliptic curves. The ECDCP problem can be stated as follows: given an elliptic curve defined over $F$ that needs N-bits for the representation of its elements, an elliptic curve point $(x_0,y_0)$ in $E(C)$, defined in affine coordinates, and a point $(x_0,y_0)$ in $E(C)$, defined in affine coordinates, determine the integer $k_0 \leq k \leq N$, such that $(x_0,y_0) = k(x_0,y_0)$, provided that such an integer exists. In the below, it is assumed that such an integer exists.

The most well known attack used against the ECDLP is the Pollard $\rho$-method, which has a complexity of $O(\sqrt{N}/\sqrt{2})$, where $K$ is the order of the underlying group, and the complexity is measured in terms of an elliptic curve point addition.

Since the underlying cryptographic problems are solved in the above block cipher chaining methods is the discrete logarithm problem, which is a known difficult mathematical problem, it is expected that the security of the above methods are more robust.

In this context, the use of the elliptic polynomial discrete logarithmic problem in the above block cipher chaining methods is the discrete logarithm problem, which is a known difficult mathematical problem, it is expected that the security of the above methods are more robust.
secure than prior art ciphers which are not based on such a
mathematically difficult problem.

Projective coordinate can also be used by the sending cor-
respondent and the receiving correspondent to embed extra
message data bits in the projective coordinate wherein the
addition of the corresponding elliptic points is defined in
(nx+ny+3) dimensional space where there are (nx+1) x-coor-
dinates, (ny+1) y-coordinates and one projective coordinate.

The equations for the addition rule can be obtained by
using the elliptic curve equation with (nx+1) x-coordinates
and (nx+1) y-coordinates in projective coordinates and substi-
tuting a straight line equation to obtain a cubic equation in
terms of one of the x-coordinates. This cubic equation can be
used to identify the third point of intersection between a
straight line and the elliptic curve in (nx+ny+3) dimensions
given two other intersection points. This third point of inter-
section is used to identify the sum of the given two points.

It should be understood that the calculations may be per-
formed by any suitable computer system, such as that dia-
grammatically shown in the sole drawing FIGURE. Data is
entered into system 100 via any suitable type of user interface
116, and may be stored in memory 112, which may be any
suitable type of computer readable and programmable
memory. Calculations are performed by processor 114, which
may be any suitable type of computer processor and may be
displayed to the user on display 118, which may be any
suitable type of computer display.

Processor 114 may be associated with, or incorporated
into, any suitable type of computing device, for example,
a personal computer or a programmable logic controller. The
display 118, the processor 114, the memory 112 and any
associated computer readable recording media are in com-
unication with one another by any suitable type of data bus,
as is well known in the art.

Examples of computer-readable recording media include a
magnetic recording apparatus, an optical disk, a magneto-
opical disk, and/or a semiconductor memory (for example,
RAM, ROM, etc.). Examples of magnetic recording apparatus
that may be used in addition to memory 112, or in place of
memory 112, include a hard disk drive (HDD), a flexible
disk (FD), and a magnetic tape (MT). Examples of the optical
disk include a DVD (Digital Versatile Disc), a DVD-RAM, a
CD-ROM (Compact Disc-Read Only Memory), and a CD-R
(Recordable)RW.

It is to be understood that the present invention is not
limited to the embodiments described above, but com-
passes any and all embodiments within the scope of the
following claims.

We claim:

1. A computerized method of performing elliptic poly-
nomial cryptography with elliptic polynomial hopping, com-
prising the steps of:

   a) defining a maximum block size that can be embedded
into (nx+1) x-coordinates and ny y-coordinates, wherein
n is an integer, and setting the maximum block size to be
(nx+ny+1)N bits, wherein N is an integer;

   b) sending a corresponding and a receiving correspond-
ent agree upon the values of nx and ny, and further agree
on a set of coefficients a,b ∈ F, wherein F represents a finite
field where the field’s elements can be represented in
N-bits, the sending and receiving correspondents further
agreeing upon a random number k_r, which is at least part
of a shared secret key for communication, a random
number generator, a random number k_p, that is a portion
of the shared secret key used for communication, and a set of
(nx+1) numbers such that x_b ∈ F, wherein i and nx

are integers, wherein i=0, . . . , nx, the set being used to find
an initial base point, the set being made public;

   the sending correspondent then performs the following
steps:

   c) generating at least a portion of a set of coefficients
b_1(0), b_2(0) ∈ F of an initial elliptic polynomial to be used
for a message authentication code of a 0-th message
block from the shared secret key k_p;

   d) embedding the set of (nx+1) numbers x_b ∈ F
into an elliptic polynomial point to obtain an initial base point
(x_0, y_0, x_1, y_1, . . . , x_n, y_n, x_y, y_y) using a data
embedding method;

   e) embedding the 0-th block of the message bit string into
an elliptic polynomial message point (x_{1, n}, y_{1, n},
. . . , x_{n, y}, y_{n, y}) using the data
embedding method;

   f) computing a scalar multiplication of the 0-th block
shared key k_b with the base point (x_0(0), y_0(0), x_1(0),
. . . , x_y(0), y_y(0)),

   g) computing a cipher point of the 0-th data block (x_{0, y}
(0), y_{0, y}(0),
. . . , x_{n, y}(0), y_{n, y}(0))

   h) generating a random number k_p with the random
number generator based upon k_p;

   j) generating at least a portion of the coefficients b_1(0),
b_2(0) ∈ F of the j-th elliptic polynomial from the random
number k_p;

   k) embedding the j-th block of the message bit string into
a j-th elliptic polynomial message point (x_{0, y}(j),
\ldots, x_{n, y}(j), y_{0, y}(j), \ldots, y_{n, y}(j)) using the data
embedding method;

   l) hopping the scalar multiplication point (x_{0, y}(j),
\ldots, x_{n, y}(j), y_{0, y}(j), \ldots, y_{n, y}(j)) to an equivalent
message authentication code point (x_{0, y}(j), y_{0, y}(j),
\ldots, x_{n, y}(j), y_{n, y}(j)) that satisfies the j-th elliptic poly-
nomial of step j) using the data embedding method;

   m) computing the cipher point of the j-th data block (x_{0, y}
(0), x_{1}(0),
\ldots, x_{n, y}(0), y_{0}(0), \ldots, y_{n, y}(0)) as:

   n) sending the cipher point (x_{0, y}(0), x_{1}(0),
\ldots, x_{n, y}(0), y_{0}(0), \ldots, y_{n, y}(0)) being sent to the receiving
 correspondent;

   r) the receiving correspondent then performs the following
steps:

   o) generating at least a portion of the coefficients b_1(0),
b_2(0) ∈ F of an initial elliptic polynomial to be used
for the message authentication code of the 0-th message
block from the shared secret key k_p;

   p) embedding the set of (nx+1) numbers x_b ∈ F
into an elliptic polynomial point to obtain an initial base point
(x_0(0), y_0(0), x_1(0), y_1(0), . . . , x_{n, y}(0), y_{n, y}(0)),

   q) computing the scalar multiplication of the 0-th block
shared key k_b with the base point (x_0(0), y_0(0), x_1(0),
A computer software product that includes a non-transitory storage medium readable by a processor, the non-transitory storage medium having stored thereon a set of instructions for performing elliptic polynomial cryptography with elliptic polynomial hopping, the instructions comprising:

a) first sequence of instructions which, when executed by the processor, causes the processor to define a maximum block size that can be embedded into (nx+1) x-coordinates and ny y-coordinates, wherein n is an integer, and setting the maximum block size to be (nx+ny+1)N bits, wherein N is an integer;

b) a second sequence of instructions which, when executed by the processor, causes the processor to implement a sending correspondent and a receiving correspondent agreeing upon the values of nx and ny, and further agreeing on a set of coefficients a,b\in\mathbb{E}$, wherein F represents a finite field where the field's elements can be represented in N-bits;

c) a third sequence of instructions which, when executed by the processor, causes the processor to implement the sending and receiving correspondents agreeing upon a random number k, which is at least part of a shared secret key for communication, and further agreeing upon a random number generator, a random number kp, that is a portion of the shared secret key used for communication, and a set of (nx+1) numbers such that xb\in\mathbb{E}$, wherein i and nx are integers, wherein i=0,\ldots, nx, the set being used to find an initial base point, the set being made public;

d) a fourth sequence of instructions which, when executed by the processor, causes the processor to generate at least a portion of a set of coefficients b_{1(i)},b_{2(i)}\in\mathbb{E}$ of a first elliptic polynomial to be used for a message authentication code of a 0-th message block from the shared secret key kp;

e) a fifth sequence of instructions which, when executed by the processor, causes the processor to embed the set of (nx+1) numbers xb\in\mathbb{E}$ into an elliptic polynomial point to obtain an initial base point (xb,xb),\ldots, (xb,xb)\in\mathbb{E}$ using the data embedding method;

f) a sixth sequence of instructions which, when executed by the processor, causes the processor to embed the 0-th block of the message bit string into an elliptic polynomial message point (x_{i0},x_{i0},\ldots,x_{i0},y_{i0},\alpha_{i0})\in\mathbb{E}$ using the data embedding method;

g) a seventh sequence of instructions which, when executed by the processor, causes the processor to compute a scalar multiplication of the 0-th block shared key k0 with the base point (x_{i0},x_{i0},\ldots,x_{i0},y_{i0},\alpha_{i0})\in\mathbb{E}$ as (x_{i0},x_{i0},\ldots,x_{i0},y_{i0},\alpha_{i0},1)\in\mathbb{E}$

h) an eighth sequence of instructions which, when executed by the processor, causes the processor to compute a cipher point of the 0-th data block (x_{i0},x_{i0},\ldots,x_{i0},y_{i0},\alpha_{i0})\in\mathbb{E}$ as:

\begin{align*}
(x_{i0},x_{i0},\ldots,x_{i0},y_{i0},\alpha_{i0})\in\mathbb{E}
\end{align*}
generate at least a portion of the coefficients $b_{1,0}^{(0)}, b_{2,0}^{(0)} \in \mathbb{F}$ of the $j$-th elliptic polynomial from the random number $\kappa_1$;

j) a twelfth sequence of instructions which, when executed by the processor, causes the processor to embed the $j$-th block of the message bit string into a $j$-th elliptic polynomial message point $(x_{0,0}, x_{1,0}, \ldots, x_{n,m}, y, y_m, \alpha_m)$ using the data embedding method;

m) a thirteenth sequence of instructions which, when executed by the processor, causes the processor to loop the scalar multiplication point $(x_{0,0}^{(l-1)}, x_{1,0}^{(l-1)}, \ldots, x_{n,m}^{(l-1)}, y, y_m, \alpha_m, y_m^{(l-1)}, \alpha_m^{(l-1)})$ to an equivalent message authentication code point $(x_{0,0}^{(l)}, x_{1,0}^{(l)}, \ldots, x_{n,m}^{(l)}, y, y_m, \alpha_m, y_m^{(l)}, \alpha_m^{(l)})$ that satisfies the $j$-th elliptic polynomial;

n) a fourteenth sequence of instructions which, when executed by the processor, causes the processor to compute the cipher point of the $j$-th data block $(x_{1,0}, y, y_m, \alpha_m)$ as:

$$
(x_{0,0}^{(l)}, x_{1,0}^{(l)}, \ldots, x_{n,m}^{(l)}, y, y_m, \alpha_m) =
$$

o) a fifteenth sequence of instructions which, when executed by the processor, causes the processor to send appropriate bits of the $x$-coordinates and a bit indicating the value of $y$ of the cipher point $(x_{0,0}^{(l)}, x_{1,0}^{(l)}, \ldots, x_{n,m}^{(l)}, y, y_m, \alpha_m)$ to the receiving processor;

p) a sixteenth sequence of instructions which, when executed by the processor, causes the processor to generate at least a portion of the coefficients $b_{1,0}^{(0)}, b_{2,0}^{(0)} \in \mathbb{F}$ of an initial elliptic polynomial to be used for the message authentication code of the $0$-th message block from the shared secret key $\kappa_2$;

q) a seventeenth sequence of instructions which, when executed by the processor, causes the processor to embed the set of $(n+1)$ numbers $x_b \in \mathbb{F}$ into an elliptic polynomial point to obtain an initial base point $(x_{0,0}^{(0)}, x_{1,0}^{(0)}, \ldots, x_{n,m}^{(0)}, y, y_m, \alpha_m)$ using the data embedding method;

r) an eighteenth sequence of instructions which, when executed by the processor, causes the processor to compute the scalar multiplication of the $0$-th block shared key $\kappa_2$ with the base point $(x_{0,0}^{(0)}, x_{1,0}^{(0)}, \ldots, x_{n,m}^{(0)}, y, y_m, \alpha_m)$ as $(x_{0,0}^{(0)}, x_{1,0}^{(0)}, \ldots, x_{n,m}^{(0)}, y, y_m, \alpha_m)$;

s) a nineteenth sequence of instructions which, when executed by the processor, causes the processor to compute the message point of the $0$-th data block $(x_{0,0}, x_{1,0}, \ldots, x_{n,m}, y, y_m, \alpha_m)$ as:

$$
(x_{0,0}^{(0)}, x_{1,0}^{(0)}, \ldots, x_{n,m}^{(0)}, y, y_m, \alpha_m) =
$$

wherein $\alpha_m = \alpha_m^{(0)}$, and recovering the secret message bit string for the $0$-th block from appropriate $x$-coordinates of the message point $(x_{0,0}^{(0)}, x_{1,0}^{(0)}, \ldots, x_{n,m}^{(0)}, y, y_m, \alpha_m)$ if $\alpha_m = 1$, and recovering the secret message bit string from a point $(g, x_{0,0}, g x_{1,0}, \ldots, g x_{n,m}, y, y_m, \alpha_m)$ if $\alpha_m = \bar{g}$;

t) a twentieth sequence of instructions which, when executed by the processor, causes the processor to initialize the integer $j$ as $j=1$ and incrementing $j$ at each step until all of the message data blocks are processed;

u) a twenty-first sequence of instructions which, when executed by the processor, causes the processor to generate a random number $\kappa_2$, with the random number generator based upon $\kappa_2$;

v) a twenty-second sequence of instructions which, when executed by the processor, causes the processor to generate at least a portion of the coefficients $b_{1,0}^{(0)}, b_{2,0}^{(0)} \in \mathbb{F}$ of the $j$-th elliptic polynomial from the random number $\kappa_2$;

w) a twenty-third sequence of instructions which, when executed by the processor, causes the processor to loop the scalar multiplication point $(x_{0,0}, x_{1,0}, \ldots, x_{n,m}, y, y_m, \alpha_m)$ of an initial elliptic polynomial to an equivalent message authentication code point $(x_{0,0}^{(l)}, x_{1,0}^{(l)}, \ldots, x_{n,m}^{(l)}, y, y_m, \alpha_m, y_m^{(l)}, \alpha_m^{(l)})$ that satisfies the $j$-th elliptic polynomial;

x) a twenty-fourth sequence of instructions which, when executed by the processor, causes the processor to compute the message point of the $j$-th received data block $(x_{0,0}, x_{1,0}, \ldots, x_{n,m}, y, y_m, \alpha_m)$ as:

$$
(x_{0,0}^{(l)}, x_{1,0}^{(l)}, \ldots, x_{n,m}^{(l)}, y, y_m, \alpha_m) =
$$

wherein $\alpha_m = \alpha_m^{(0)}$; and

y) a twenty-fifth sequence of instructions which, when executed by the processor, causes the processor to recover the secret message bit string for the $j$-th block from appropriate $x$-coordinates of the message point $(x_{0,0}, x_{1,0}, \ldots, x_{n,m}, y, y_m, \alpha_m)$ if $\alpha_m = 1$, and recovering the secret message bit string from the point $(g, x_{0,0}, g x_{1,0}, \ldots, g x_{n,m}, y, y_m, \alpha_m)$ if $\alpha_m = \bar{g}$.