

Non-Replicable Function-Based Multi-Level Random Dynamic Secret Key Algorithm (Extended Version)

Ming Xian¹, Xunwei Zhou^{2*} and Qinguang Chen³

¹Fushun Big Data Management Center, Fushun, Liaoning, China

²Beijing Key Laboratory of Information Service Engineering, Beijing Union University Beijing, China

³Puruite Printing House Fushun, Liaoning, China

*Corresponding author

Xunwei Zhou, Beijing Key Laboratory of Information Service Engineering, Beijing Union University Beijing, China.

Received: June 29, 2026; Accepted: July 06, 2026; Published: July 13, 2026

ABSTRACT

Based on the proof that, as an encryption function, the residual function $y=(ax)\bmod m$ is a non-replicable function, this paper constructs multi-level encryption functions $y_1=(a_1x)\bmod m_1$, $y_n=(a_nx)\bmod m_n$. And based on this the paper constructs the multi-level random dynamic secret key algorithm XM-1. XM-1 has three-fold safeguards. First, the encryption function is not replicable. Secondly, the secret keys and the number of secret keys is randomized. Thirdly, when the system does not run, there is no secret key. After slight changes, the four examples for constructing XM-1 given in this paper can be put into practical use.

Keywords: Residual Function, Not Replicable Function, Multi-Level Encryption Function, XM-1

Introduction

The facts attest that, there is no prospect to achieve the safety of the secret key systems by enlarging the secret key space and raising the computational complexity. Because the explosive increase of the computing power make this effort futile. Another challenge is that the network attack is everywhere. The secret key in the system being unchanged for a long time is a hidden peril. Facing the first challenge, we need to find a function that is not an encryption function obtained from plain texts and cipher texts by mathematical methods. While $y=(ax)\bmod m$ is just this kind of function. Facing the second challenge, we need to construct a secret key system that does not have a secret key when the system does not run.

XM-1 is Just this Kind of System

The rest of this paper is organized as follows: In Section 2 the residual function and the non-replicable function are introduced. Section 3 is about the conditions to be met for multi-level residual functions as encryption functions. In Section 4 the construction of the multi-level random dynamic secret key system is introduced. Section 5 deals with the elimination of

duplicate cipher texts. Section 6 is the discussion of practical program.

Residual Function and not Replicable Function

According to the property of the congruent formulas, $a\equiv r \pmod m$ can be denoted by $a=km+r$ (k is an integer). When $0\leq r<m$, r is called the least non-negative residue of a module m . For convenience, we denote the least non-negative residue of a module m as $(a)_{\bmod m}$.

Definition 1: $r=(a)_{\bmod m}$ if and only if r is the least non-negative residue of a module m .

From Definition 1 we obtain:

Property 1: $0\leq (a)_{\bmod m} < m$.

For example, $(9)_{\bmod 7}=2<7$, $(78)_{\bmod 35}=8<35$.

Property 2: If $r=(a)_{\bmod m}$ then $r\equiv a \pmod m$.

Property 3: If $r=(a)_{\bmod m}$ then $a=km+r$ (k is an integer).

Property 4: Suppose $a\equiv b \pmod m$. If $0\leq a<m$ and $0\leq b<m$ then $a=b$.

Property 5: If $a \equiv b \pmod{m}$ then $(a) \bmod m \equiv b \pmod{m}$.

Now, we investigate:

$$y = (ax)_{\bmod m} \quad \square \square \square$$

It is not hard to discover that, when a, m in formula (1) are parameters, then for every x there necessarily exists a unique y corresponding to it.

For example, in the case of $x=5, m=17, y=(15x) \bmod 17 = (15 \times 5)_{\bmod 17} = 7$ uniquely. Thus, we have

Conclusion 1: In formula (1), y is the function of x .

We call the function in formula (1) as y being the least residual function or residual function of x .

Definition 2: When $(a, m) = 1$, we call (a, m) as a parameter couple. When a^{-1} is the inverse of a module m , then we call (a^{-1}, m) as the inverse couple of (a, m) . If parameter couple A is the inverse couple of parameter couple B , then we say that A and B are mutually inverse parameter couples. Suppose (a_i, m_i) are parameter couples (where $i=1, n$). When $n>1$, we call $(a_1, m_1), \dots, (a_n, m_n)$ as n -level parameter queue. And we call $(a_1^{-1}, m_1), \dots, (a_n^{-1}, m_n)$ as the inverse queue of $(a_1, m_1), \dots, (a_n, m_n)$. If n -level parameter queue A is the inverse queue of n -level parameter queue B , then we call A and B as the mutually inverse n -level parameter queues.

Theorem 1: Suppose $(a, m) = 1$. If $y = (ax)_{\bmod m}, 0 \leq x < m$, then $x = (a^{-1}y)_{\bmod m}$.

Proof: From $(a, m) = 1$ we know that, there necessarily exists a^{-1} which is the inverse of a module m .

From $y = (ax)_{\bmod m}$ and Property 2 we know that, $y \equiv ax \pmod{m}$. Hence, $a^{-1}y \equiv x \pmod{m}$.

From Property 5 we know that, $(a^{-1}y)_{\bmod m} \equiv x \pmod{m}$.

Besides, from Property 1 we know that, $0 \leq (a^{-1}y)_{\bmod m} < m$. And from the supposition, we have, $0 \leq x < m$.

Thus, from Property 4 we have, $x = (a^{-1}y)_{\bmod m}$. Q.E.D.

From Theorem 1 we know that, when we take $x(x < m)$ as the plain text, by $y = (ax)_{\bmod m}$ we can obtain cipher text y ; from cipher text y by $x = (a^{-1}y)_{\bmod m}$ we can obtain plain text x .

Obviously, (a, m) --the corresponding parameter couple of $y = (ax)_{\bmod m}$ --and (a^{-1}, m) --the corresponding parameter couple of $x = (a^{-1}y)_{\bmod m}$ --are mutually inverse parameter couples.

Example 1: It is known that $(53, 264)$ and $(5, 264)$ are mutually inverse parameter couples.

$x=23$ is a plain text. By $y = (53x)_{\bmod 264}$ we can obtain the cipher text $y = (53 \times 23)_{\bmod 264} = 163$.

$y=163$ is a cipher text. By $x = (5y)_{\bmod 264}$ we can obtain the plain text $x = (5 \times 163)_{\bmod 264} = 23$.

Because $\begin{cases} (53, 264) \\ (5, 264) \end{cases}$ and $\begin{cases} y = (53x)_{\bmod 264} \\ x = (5y)_{\bmod 264} \end{cases}$ are one to one correspondent, we use the former to replace the latter.

Example 2: When $\begin{cases} (53, 264) \\ (5, 264) \end{cases}$ are mutually inverse parameter couples, then from $(53, 264)$ and plain text 23 we can obtain cipher text $(53 \times 23)_{\bmod 264} = 163$. And from $(5, 264)$ and cipher text 163 we can obtain plain text $(5 \times 163)_{\bmod 264} = 23$.

Besides, from $(5, 264)$ and plain text 23 we can obtain cipher text $(5 \times 23)_{\bmod 264} = 115$. And from $(53, 264)$ and cipher text 115 we can obtain plain text $(53 \times 115)_{\bmod 264} = 23$.

This means that, when A side has (a, m) and B side has (a^{-1}, m) , then A side and B side are mutual encryption-decryption sides. The cipher text obtained from A side by (a, m) can be decrypted by B side; the cipher text obtained from B side by (a^{-1}, m) can be decrypted by A side.

Conclusion 2: If A side has (a, m) and B side has (a^{-1}, m) , then A side and B side are mutual encryption-decryption sides.

Now, we investigate the importance of condition " $x < m$ " in **Theorem 1**.

We still use $y = (53x)_{\bmod 264}$ as the encryption function, use $x = (5y)_{\bmod 264}$ as the decryption function.

$x=287$ is a plain text. By $y = (53x)_{\bmod 264}$ we can obtain the cipher text $y = (53 \times 287)_{\bmod 264} = 163$.

$y=163$ is the cipher text. By $x = (5y)_{\bmod 264}$ we obtain $x = (5 \times 163)_{\bmod 264} = 23$. $23 \neq 287$, i.e. 23 is not the plain text.

The reason is simple: the plain text $x=287 > 264$, which do not satisfy the condition " $x < m$ ".

Now, we define replicable functions and non-replicable functions. First, we expound the reason for making the definitions.

Let us investigate the relationship among x, y and parameters a_1, a_2 in formula:

$$y = (a_1x)^2 + a_2 \quad \square \square \square$$

Suppose $x=2, y=17$ and $x=3, y=37$ are two couples of assignments to formula (2).

Substituting $x=2, y=17$ into formula (2) we obtain $17 = 4a_1^2 + a_2$. Substituting $x=3, y=37$ into formula (2) we obtain $37 = 9a_1^2 + a_2$.

Then we obtain the following system of linear equations of two unknowns: $\begin{cases} 17 = 4a_1^2 + a_2 \\ 37 = 9a_1^2 + a_2 \end{cases}$

Solving this system, we can obtain $a_1 = \pm 2, a_2 = 1$. Substituting them into formula (2) we can obtain $y = (\pm 2x)^2 + 1$.

Please look back. The two couples of assignments $x=2, y=17$ and $x=3, y=37$ are originally obtained from $y = (2x)^2 + 1$ (or $y = (-2x)^2 + 1$). Thus, when we suppose $y = (2x)^2 + 1$ (or $y = (-2x)^2 + 1$) is an

encryption function, the above process is one from plain texts 2,3 and cipher texts 17,37 through $y=(a_1x)^2+a_2$ to replicate the encryption function.

For convenience, we call $y=(a_1x)^2+a_2$ as the object function, call $y=(2x)^2+1$ (or $y=(-2x)^2+1$) as the candidate function. Obviously, from the two couples of assignments $x=2, y=17$ and $x=3, y=37$ we can only obtain "2" candidate functions: $y=(2x)^2+1$ and $y=(-2x)^2+1$. This is to say, the number of candidate functions obtained through the object function is definite.

Definition 3: Suppose there are n parameters in function $y=f(x)$. Also suppose from the n couples of x, y through object function $y=f(x)$ we can obtain k candidate functions. If k , the number of candidate functions obtained, is definite, then we call $y=f(x)$ as a replicable function, otherwise, we call $y=f(x)$ as a non-replicable function.

Obviously, $y=(a_1x)^2+a_2$ is a replicable function. It is not hard to discover that, general functions are all replicable functions.

Yet, $y=(ax) \bmod m$ is not a replicable function.

There are two parameters in $y=(ax) \bmod m$: a, m .

Suppose $x=14, y=58$ and $x=25, y=47$ are the two couples of assignments to $y=(ax) \bmod m$.

From $x=14, y=58$ and $y=(ax) \bmod m$ we obtain, $58=(14a) \bmod m$. By Property 3 we obtain, $14a=k_1m+58$.

From $x=25, y=47$ and $y=(ax) \bmod m$ we obtain, $47=(25a) \bmod m$. By Property 3 we obtain, $25a=k_2m+47$.

Thus, we have the following system of equations:
$$\begin{cases} 14a = k_1m + 58 \\ 25a = k_2m + 47 \end{cases}$$

Solving the system, we obtain $m=264, a=23$. Thus, from $y=(ax) \bmod m$ we obtain: $y=(23x) \bmod 264$.

Likewise, let $k_1=4, k_2=8$, we obtain $m=66, a=23$. Thus, $y=(23x) \bmod 66$.

Let $k_1=5, k_2=9$, we obtain $m=792, a=287$. Thus, $y=(287x) \bmod 792$.

Please verify: $x=14, y=(23x) \bmod 264 = (23x) \bmod 66 = (287x) \bmod 792 = 58$.

$x=25, y=(23x) \bmod 264 = (23x) \bmod 66 = (287x) \bmod 792 = 47$.

That is to say, functions $y=(23x) \bmod 264, y=(23x) \bmod 66, y=(287x) \bmod 792$, satisfy $x=14, y=58$ and $x=25, y=47$.

Besides, it is not hard to discover that $y=(23x) \bmod 264$ and $y=((23+264n)x) \bmod 264$ (n is a positive integer) are the functions with the same value (Because $(23x) \bmod 264 = (23x+264nx) \bmod 264 = ((23+264n)x) \bmod 264$), $y=(23x) \bmod 66$ and $y=((23+66n)x) \bmod 66$ are the functions with the same value, $y=(287x) \bmod 792$ and $y=((287+792n)x) \bmod 792$ are the functions with the same value.

This means that there are infinitely many functions $y=((23+264n)x) \bmod 264, y=((23+66n)x) \bmod 66, y=((287+792n)x) \bmod 792$, satisfying the two couples of assignments $x=14, y=58$ and $x=25, y=47$.

The other side of the facts is that, $y=(23x) \bmod 264, y=(23x) \bmod 66, y=(287x) \bmod 792$ are different functions.

For example, when $x=15, y=(23x) \bmod 264 = (23 \times 15) \bmod 264 = 81; y=(23x) \bmod 66 = (23 \times 15) \bmod 66 = 15; y=(287x) \bmod 792 = (287 \times 15) \bmod 792 = 345$.

We have discussed the case of two couples of assignments of x, y to $y=(ax) \bmod m$ above. Now, let us investigate the case of infinitely many couples of assignments of x, y to $y=(ax) \bmod m$.

Now, suppose $x=x_1, x_n$, by $y=(ax) \bmod m$ we can obtain $y=y_1, y_n$.

Thus,
$$\begin{cases} ax_1 = k_1m + y_1 \\ ax_n = k_nm + y_n \end{cases}$$

There are n equations in this system, but $n+2$ unknowns: a, m, k_1, k_n . Therefore, the system still has infinitely many solutions. This means,

Conclusion 3: No matter how many plain texts x and cipher texts y are known, the finite number of the encryption functions in the form of $y=(ax) \bmod m$ cannot be obtained.

Conclusion 3 tells us that, the encryption functions in the form of $y=(ax) \bmod m$ are non-replicable. Because the process of deciphering the secret key systems can be simply regarded as one of replicating the encryption functions, that the encryption functions cannot be replicated means that the secret key systems based on them are non-replicable.

The Conditions to be Met for Multi-Level Residual Functions as Encryption Functions

Now, we discuss the conditions to be met for

$$y_1=(a_1x) \bmod m_1, y_n=(a_nx) \bmod m_n \quad \square\square\square$$

being the encryption functions. For this reason, we call y_n in formula (3) as the n -level residual function. When $n>1$, we call y_n as the multi-level residual function of x .

Through the discussion in Section 2 we know that, when we take a_1, a_n, m_1, m_n as the secret keys, from plain text x by non-replicable function $y_1=(a_1x) \bmod m_1$ we can obtain cipher text y_1 . We call y_1 as 1-level cipher text of x .

Likewise, from 1-level cipher text y_1 by non-replicable function $y_2=(a_2y_1) \bmod m_2$ we can obtain 2-level cipher text y_2 , from $(n-1)$ -level cipher text y_{n-1} by non-replicable function $y_n=(a_ny_{n-1}) \bmod m_n$ we can obtain n -level cipher text y_n .

From this we know that, formula (3) are the functions that can perform multi-level encryption process. Obviously, the secret key space is infinity. (Interested readers can verify yourselves) Now, we discuss the condition to be met for formula (3) being the encryption functions.

Theorem 2: I. If in formula (3), $x < m_1, y_1 < m_2, y_{n-1} < m_n$, then $y_{n-1}=(a_{n-1}y_n) \bmod m_n, \dots, x=(a_1^{-1}y_1) \bmod m_1$ $\square\square\square$

II. If in formula (3), $x < m_1 < m_2 \dots < m_n$, then in formula (3), $x < m_1, y_1 < m_2, y_{n-1} < m_n$.

Proof: Prove I. From $y_n = (a_n y_{n-1})_{\text{mod } mn}$ in formula (3) and Property 2 we know that, $y_n \equiv a_n y_{n-1} \pmod{m_n}$.

Thus, $y_{n-1} \equiv a_n^{-1} y_n \pmod{m_n}$. From the supposition $y_{n-1} < m_n$ and $a_n^{-1} y_n \pmod{m_n} < mn$ and Property 4 we know that, $y_{n-1} = (a_n^{-1} y_n)_{\text{mod } mn}$.

Likewise, we can prove, $y_{n-2} = (a_{n-1}^{-1} y_{n-1})_{\text{mod } mn-1}, \dots, x = (a_1^{-1} y_1)_{\text{mod } m1}$.

Prove II. We only prove that, if $m_1 < m_2 \dots < mn$ then $y_1 < m_2, y_{n-1} < m_n$.

From $y_1 = (a_1 x)_{\text{mod } m1}$ in formula (3) and Property 1 we know that, $y_1 < m_1$. From the supposition $m_1 < m_2$ we know that, $y_1 < m_2$.

Likewise, we can prove that, $y_2 < m_3, y_{n-1} < m_n$. Q.E.D.

It is not hard to discover that, the subscripts of “m” in formula (3) are from 1 to n, while those in formula (4) are from n to 1. Now, we make the following arrangement:

$$\begin{array}{ll} y_1 = (a_1 x)_{\text{mod } m1}, y_n = (a_n y_{n-1})_{\text{mod } mn} & \square\square\square \\ x = (a_1^{-1} y_1)_{\text{mod } m1}, y_{n-1} = (a_{n-1}^{-1} y_n)_{\text{mod } mn} & \square\square\square \end{array}$$

At this time, the process of obtaining x from y_n needs to start from $y_{n-1} = (a_n^{-1} y_n)_{\text{mod } mn}$ at the right of formula (6), going leftward, and to terminate at $x = (a_1^{-1} y_1)_{\text{mod } m1}$.

Besides, we draw out the parameters of formulas (5) and (6) to obtain the following mutually inverse n-level parameter queues:

$$\begin{array}{ll} (a_1, m_1), (a_n, m_n) & \square\square\square \\ (a_1^{-1}, m_1), (a_n^{-1}, m_n) & \square\square\square \end{array}$$

The process of obtaining y_n from x by formula (5) can be simplified as one by formula (7). At this time, the operations are from left to right, we call this process as the operations starting from the left. The process of obtaining x from y_n by formula (6) can be simplified as one by formula (8). At this time, the operations are from right to left, we call this process as the operations starting from the right.

Example 3: It is known: $\{(61,93), (13,155), (25,231) \dots (61,93), (12,155), (37,231) \dots$ are mutually inverse 3-level parameter queues. We use two methods to perform the encryption process for plain text 23.

Method 1

The encryption process—Performing the operations starting from the left using the front queue $(61,93)_1, (13,155)_2, (25,231)_3$ (The subscripts of the parameter couples are their series numbers) to 23 obtains the cipher text:

From 23 and $(61,93)_1$ we obtain $(61 \times 23)_{\text{mod } 93} = 8$, from 8 and $(13,155)_2$ we obtain $(13 \times 8)_{\text{mod } 155} = 104$, from 104 and $(25,231)_3$ we obtain $(25 \times 104)_{\text{mod } 231} = 59$. That is, the cipher text is 59.

The decryption process—Performing the operations starting from the right using the rear queues $(61,93)_1, (12,155)_2, (37,231)_3$ to 59 obtains the plain text:

From 59 and $(37,231)_3$ we obtain $(37 \times 59)_{\text{mod } 231} = 104$, from 104 and $(12,155)_2$ we obtain $(12 \times 104)_{\text{mod } 155} = 8$, from 8 and $(61,93)_1$ we obtain $(61 \times 8)_{\text{mod } 93} = 23$.

Method 2

The encryption process—Performing the operations starting from the left using rear queue $(61,93), (12,155), (37,231)$ to 23 we obtain the cipher text:

From 23 and $(61,93)$ we obtain $(61 \times 23)_{\text{mod } 93} = 8$, from 8 and $(12,155)$ we obtain $(12 \times 8)_{\text{mod } 155} = 96$, from 96 and $(37,231)$ we obtain $(37 \times 96)_{\text{mod } 231} = 87$. That is, the cipher text is 87.

The decryption process—Performing the operations starting from the right using front queue $(61,93), (13,155), (25,231)$ to 87 we obtain the plain text:

From 87 and $(25,231)$ we obtain $(25 \times 87)_{\text{mod } 231} = 96$, from 96 and $(13,155)$ we obtain $(13 \times 96)_{\text{mod } 155} = 8$, from 8 and $(61,93)$ we obtain $(61 \times 8)_{\text{mod } 93} = 23$.

Example 3 tells us that, performing the operations starting from the left using formula (7) (or formula (8)) to the plain text we can obtain the cipher text; performing the operations starting from the right using formula (8) (or formula (7)) to the cipher text we can obtain the plain text.

Now, we discuss condition “ $x < m_1, y_1 < m_2, y_{n-1} < m_n$ ” in **Theorem 2**.

There are two cases need to be considered. One is the case of the condition “ $m_1 < m_2 < m_n$ ” being satisfied (See Example 3, because $93 < 155 < 231$) The other is the case of the condition “ $m_1 < m_2 < m_n$ ” not being satisfied (See Example 4).

Example 4: Use the mutually inverse 3-level parameter queues $\{(13,155), (61,93), (25,231) \dots (12,155), (61,93), (37,231) \dots$ to perform the encryption-decryption processes to plain texts 19 and 23.

1. The encryption-decryption processes to plain text 19.

The encryption process—using $(13,155), (61,93), (25,231)$ performing the operations starting from the left to 19 obtains the cipher text.

From 19 and $(13,155)$ we obtain $(13 \times 19)_{\text{mod } 155} = 92$ (It is y_1), from 92 and $(61,93)$ we obtain $(61 \times 92)_{\text{mod } 93} = 32$ (It is y_2), from 32 and $(25,231)$ we obtain $(25 \times 32)_{\text{mod } 231} = 107$ (It is y_3). The cipher text is 107.

At this time, $x = 19 < 155, y_1 = 92 < 93, y_2 = 32 < 231$. That is, “ $x < m_1, y_1 < m_2, y_{n-1} < m_n$ ” hold.

The decryption process—using $(12,155), (61,93), (37,231)$ performing the operations starting from the right to $y_3 = 107$ obtains x.

From 107 and $(37,231)$ we obtain $y_2 = (37 \times 107)_{\text{mod } 231} = 32$, from 32 and $(61,93)$ we obtain $y_1 = (61 \times 32)_{\text{mod } 93} = 92$, from 92 and $(12,155)$ we obtain $x = (12 \times 92)_{\text{mod } 155} = 19$. 19 is just the plain text given.

The above facts show that, when $y_1 = (a_1 x)_{\text{mod } m1}, y_n = (a_n y_{n-1})_{\text{mod } mn}, x < m_1, y_1 < m_2, y_{n-1} < mn$ are the sufficient conditions for $y_{n-1} = (a_{n-1}^{-1} y_n)_{\text{mod } mn}, \dots, x = (a_1^{-1} y_1)_{\text{mod } m1}$ to hold.

2. The encryption-decryption processes to plain text 23.

The encryption process—using (13,155), (61,93), (25,231) performing the operations starting from the left to $x=23$ obtains y_3 .

From 23 and (13,155) we obtain $(13 \times 23)_{\text{mod } 155} = 144$ (It is y_1), from 144 and (61,93) we obtain $(61 \times 144)_{\text{mod } 93} = 42$ (It is y_2), from 42 and (25,231) we obtain $(25 \times 42)_{\text{mod } 231} = 126$ (It is y_3).

At this time, $y_1=144>93$, i.e., “ $y_1>m_2$ ”. Thus, “ $x>m_1$, $y_1>m_2$, $y_n>m_n$ ” do not hold.

The decryption process—using (12,155), (61,93), (37,231) performing the operations starting from the right to $y_3=126$ obtains x .

From 126 and (37,231) we obtain $y_2 = (37 \times 126)_{\text{mod } 231} = 42$, from 42 and (61,93) we obtain $y_1 = (61 \times 42)_{\text{mod } 93} = 51$, from 51 and (12,155) we obtain $x = (12 \times 51)_{\text{mod } 155} = 150$, it is not plain text 23.

Example 4 shows that, when $y_1=(a_1x)_{\text{mod } m_1}$, $y_n=(a_n y_{n-1})_{\text{mod } m_n}$, upon the condition $x<m_1$, $y_1<m_2$, $y_{n-1}<m_n$, not all x ($x<m_1$) can be obtained from $y_{n-1}=(a_n^{-1}y_n)_{\text{mod } m_n}$, ..., $x=(a_1^{-1}y_1)_{\text{mod } m_1}$. Only the condition $x<m_1<m_2<\dots<m_n$ is satisfied, can formula (3) be an encryption function.

Besides, when $(a_i, m_i)=1$, if there exists $(a_1, m_1), \dots, (a_n, m_n)$, then there necessarily exists $(a_1^{-1}, m_1), (a_n^{-1}, m_n)$. Thus, according to the above discussion, we have:

Conclusion 4: Suppose $m_1<m_2<\dots<m_n$. If A side has $(a_1, m_1), \dots, (a_n, m_n)$, B side has $(a_1^{-1}, m_1), (a_n^{-1}, m_n)$, then A side and B side are encryption-decryption sides mutually.

We call $(a_1, m_1), \dots, (a_n, m_n)$ or $(a_1^{-1}, m_1), (a_n^{-1}, m_n)$ in Conclusion 4 as the n -level parameter queues agreeing with the conditions.

The Construction of the Multi-Level Random Dynamic Secret Key System

Definition 4: When a secret key system is activated, the encryption-decryption sides use the same random number d_{sj} to generate parameter queues agreeing with the conditions $(a_1, m_1), \dots, (a_n, m_n)$ and $(a_1^{-1}, m_1), \dots, (a_n^{-1}, m_n)$ separately, and perform the encryption-decryption processes according to the methods given in Example 3, this secret key system is called a multi-level random dynamic secret key system, abbreviated as XM-1.

(Note: XM-1 is a secret key system. The method to construct it is calling XM-1 algorithm. So, viewed from different angle, XM-1 can denote either the system or the algorithm)

Now, we give the method for constructing XM-1. The construction of XM-1 has no fixed pattern; we can only expound it by examples for XM-1.

As Example 3 has already given the methods for the encryption-decryption using the n -level parameter queues agreeing with the conditions, in the examples we only discuss the method of A side acquiring the n -level parameter queues agreeing with the conditions $(a_1, m_1), \dots, (a_n, m_n)$ and that of B side acquiring the

n -level parameter queues agreeing with the conditions $(a_1^{-1}, m_1), \dots, (a_n^{-1}, m_n)$.

Lemma 1: If $m=ab+1$ then the inverse of a module m is $a^{-1} \equiv m-b \pmod{m}$.

Proof: From $m=ab+1$ we know that, $ab=m-1$. Then, $a(m-b) = am - ab = am - (m-1) = (a-1)m + 1$. Thus, $a(m-b) \equiv 1 \pmod{m}$, $a^{-1} \equiv m-b \pmod{m}$. Q.E.D.

Special appointments: a^{-1} is the inverse of a module m , a_n^{-1} is the inverse of a_n module m_n , $n=1, 2, \dots$

From Lemma 1 and Special appointments we obtain:

Skill 1: If $m_i=a_i b_i+1$ then $a_i^{-1}=m_i-b_i$; or if $m_i=ab_i+1$ then $a_i^{-1}=m_i-b_i$; if $m_i=a_i b+1$ then $a_i^{-1}=m_i-b$.

Skill 1 gives 3 results; their usages are different. We only give an example to expound the usage of “if $m_i=a_i b+1$ then $a_i^{-1}=m_i-b$ ”.

Let $b=17$.

Let $a_1=18$ we obtain $m_1=18 \times 17 + 1 = 307$. Thus, $a_1^{-1} = m_1 - 17 = 307 - 17 = 290$.

Let $a_2=20$ we obtain $m_2=20 \times 17 + 1 = 341$. Thus, $a_2^{-1} = m_2 - 17 = 341 - 17 = 324$.

Let $a_3=22$ we obtain $m_3=22 \times 17 + 1 = 375$. Thus, $a_3^{-1} = m_3 - 17 = 375 - 17 = 358$.

Hence, we obtain the following mutually inverse 3-level parameter queues agreeing with the conditions.

(18, 307), (20, 341), (22, 375), (290, 307), (324, 341), (358, 375).

Let $b=10$.

Let $a_1=18$ we obtain $m_1=18 \times 10 + 1 = 181$. Thus, $a_1^{-1} = m_1 - 10 = 181 - 10 = 171$.

Let $a_2=20$ we obtain $m_2=20 \times 10 + 1 = 201$. Thus, $a_2^{-1} = m_2 - 10 = 201 - 10 = 191$.

Let $a_3=22$ we obtain $m_3=22 \times 10 + 1 = 221$. Thus, $a_3^{-1} = m_3 - 10 = 221 - 10 = 211$.

Hence, we obtain the following mutually inverse 3-level parameter queues agreeing with the conditions.

(18, 181), (20, 201), (22, 221), (171, 181), (191, 201), (211, 221)

When b is random number d_{sj} , by the above process we obtain an XM-1.

Example 1 for XM-1: Use “if $m_i=a_i b+1$ then $a_i^{-1}=m_i-b$ ” to design an XM-1 with the plain text x (i.e., $0 < x < 10^8$) not exceeding decimal numbers of 8 digits.

Based on Conclusion 4, we design an XM-1 with $10^8 \leq m_1 < m_2 < \dots < m_n < 10^{10}$.

Based on the following appointments, A side obtains n -level parameter queue agreeing with the conditions $(a_1, m_1), \dots, (a_n, m_n)$:

1. If $(d_{sj})_{\text{mod } 51} < 3$ then $n=(d_{sj})_{\text{mod } 51} + 3$, otherwise $n=(d_{sj})_{\text{mod } 51}$. (That is $3 \leq n < 51$).
2. If $(d_{sj})_{\text{mod } 714285} < 88000$ then $b=(d_{sj})_{\text{mod } 714285} + 88000$, otherwise $b=(d_{sj})_{\text{mod } 714285}$. (That is $88000 \leq b < 714285$).

3. If $(b)_{\text{mod } 2400} < 1137$ then $a = (b)_{\text{mod } 2400} + 1137$, otherwise $a = (b)_{\text{mod } 2400}$ (That is $1137 \leq a < 2400$).
4. $m_1 = ab + 1, m_2 = (a+5)b + 1, \dots, m_n = (a+5(n-1))b + 1$.
5. $a_1 = a, a_2 = a_1 + 5, \dots, a_n = a_{n-1} + 5$.

(Note: The minimal $a = 1137$, the minimal $b = 88000$, the minimal $m_1 = 1137 \times 88000 + 1 = 100056001 > 10^8$)

Based on the following appointments, B side obtains n-level parameter queue agreeing with the conditions $(a_1^{-1}, m_1), (a_n^{-1}, m_n)$:

1. If $(d_{SJ})_{\text{mod } 51} < 3$ then $n = (d_{SJ})_{\text{mod } 51} + 3$, otherwise $n = (d_{SJ})_{\text{mod } 51}$.
2. If $(d_{SJ})_{\text{mod } 714285} < 88000$ then $b = (d_{SJ})_{\text{mod } 714285} + 88000$, otherwise $b = (d_{SJ})_{\text{mod } 714285}$.
3. If $(b)_{\text{mod } 2400} < 1137$ then $a = (b)_{\text{mod } 2400} + 1137$, otherwise $a = (b)_{\text{mod } 2400}$.
4. $m_1 = ab + 1, m_2 = (a+5)b + 1, m_n = (a+5(n-1))b + 1$.
5. $a_1^{-1} = m_1 - b, a_2^{-1} = m_2 - b, \dots, a_n^{-1} = m_n - b$.

(Note: The first 4 appointments of A and B sides are the same correspondingly).

Thus, a practical XM-1 has been constructed. But, at this time, we—the designer—don't know the number of secret keys of the system and we don't know any secret key either, for we don't know what dSJ is. When A side (or B side) sends random number d_{SJ} to the opposite side, XM-1 is activated.

Now, suppose A side sends random number $d_{SJ} = 629629579$ to B side. Thus, both sides has $d_{SJ} = 629629579$.

(Note: d_{SJ} can be generated either by a random function automatically, or by human beings as required. The XM-1 activated by the former is called free XM-1, that activated by the latter is called controlled XM-1. The controlled XM-1 is usually used in the special circumstances. In this paper only free XM-1 are discussed.)

A side:

According to appointment (1), from

$$(d_{SJ})_{\text{mod } 51} = (629629579)_{\text{mod } 51} = 1 < 3 \text{ we know that, } n=4.$$

According to appointment (2), from

$$(d_{SJ})_{\text{mod } 714285} = (629629579)_{\text{mod } 714285} = 344494 > 88000 \text{ we know that, } b=344494.$$

According to appointment (3), from

$$(344494)_{\text{mod } 2400} = 1294 > 1137 \text{ we know that, } a=1294.$$

According to appointment (4),

$$\begin{aligned} m_1 &= 1294 \times 344494 + 1 = 445775237, \\ m_2 &= 1299 \times 344494 + 1 = 447497707, \\ m_3 &= 1304 \times 344494 + 1 = 449220177, \\ m_4 &= 1309 \times 344494 + 1 = 450942647. \end{aligned}$$

According to appointment (5), $a_1 = 1294, a_2 = 1299, a_3 = 1304, a_4 = 1309$.

Thus, we obtain 4-level parameter queue agreeing with the conditions: $(1294, 445775237), (1299, 447497707), (1304, 449220177), (1309, 450942647)$.

B side:

According to appointment (1), from

$$(d_{SJ})_{\text{mod } 51} = (629629579)_{\text{mod } 51} = 1 < 3 \text{ we know that, } n=4.$$

According to appointment (2), from

$$(d_{SJ})_{\text{mod } 714285} = (629629579)_{\text{mod } 714285} = 344494 > 88000 \text{ we know that, } b=344494.$$

According to appointment (3), from

$$(344494)_{\text{mod } 2400} = 1294 > 1137 \text{ we know that, } a=1294.$$

According to appointment (4),

$$\begin{aligned} m_1 &= 1294 \times 344494 + 1 = 445775237, \\ m_2 &= 1299 \times 344494 + 1 = 447497707, \\ m_3 &= 1304 \times 344494 + 1 = 449220177, \\ m_4 &= 1309 \times 344494 + 1 = 450942647. \end{aligned}$$

According to appointment (5), $a_1 = 1294, a_2 = 1299, a_3 = 1304, a_4 = 1309$.

$$a_1^{-1} = 445430743, a_2^{-1} = 447153213, a_3^{-1} = 448875683, a_4^{-1} = 450598153.$$

Thus, we Obtain 4-level Parameter Queue Agreeing with the Conditions

$(445430743, 445775237), (447153213, 447497707), (448875683, 449220177), (450598153, 450942647)$.

We expound Example 1 for XM-1 as follows:

1. XM-1 is composed of the making process (or stage) and the using process (or stage). The making of XM-1 is like that of the other secret key systems; it can be accomplished by its own design department or entrust a design company. In this example, during the making process, A side only knows its own 5 appointments, B side only knows its own 5 appointments either. Therefore, during this process there is no information exchange. When XM-1 is activated, it enters the using stage.
2. The random number d_{SJ} in XM-1 can be any positive integer. This means that d_{SJ} is only a random number, it does not contain any other information. Although d_{SJ} is transmitted to the opposite side through the public information channels, it does not reveal any information of the systems. XM-1 neither transmits the public key nor exchanges the secret keys. It seems it is a private key system. Yet, it thoroughly overcomes the weaknesses of the traditional private key systems such as the private keys and the number of private keys are fixed, it is hard for the systems to be dynamic, and it suffers from the network assaults owing to the private keys being deposited in the systems for long.
3. "If $(d_{SJ})_{\text{mod } 51} < 3$ then $n = (d_{SJ})_{\text{mod } 51} + 3$, otherwise $n = (d_{SJ})_{\text{mod } 51}$ " (i.e., Appointment (1)) manifests that level n of the parameter queue agreeing with the conditions varies randomly between 3 and 50 (If necessary, it can vary randomly between 50 and 500), That is, the number of the secret keys is randomized. Obviously, the secret key systems with the number of secret keys being randomized are rare.
4. "If $(d_{SJ})_{\text{mod } 714285} < 88000$ then $b = (d_{SJ})_{\text{mod } 714285} + 88000$ " (i.e., Appointment (2)) manifests that b varies randomly between 88000 and 714284. "If $(b)_{\text{mod } 2400} < 1137$ then $a = (b)_{\text{mod } 2400} + 1137$, otherwise $a = (b)_{\text{mod } 2400}$ " (i.e., Appointment (3)) manifests that a varies randomly between 1137 and 2399. This means that the secret keys m_1, m_n in XM-1 and $a_1, a_n, a_1^{-1}, a_n^{-1}$ are highly randomized. The secret keys and the

number of secret keys is multi-fold randomized, and every level encryption functions of the multi-level encryption functions are non-replicable functions. Therefore, XM-1 cannot be attacked by brutal force, or we can say that XM-1 is unconditionally safe.

- Before and after the running of XM-1 there is no secret key, which results in that the network assaulters cannot find targets. This makes XM-1 the first choice of the network secret key systems.

Now, we continue our discussion of the other methods for making XM-1. Let $b_i = (m_i - 1)/a_i$ in Skill 1, we obtain:

Skill 2: If $m_i = a_i b_i + 1$ then $a_i^{-1} = m_i - (m_i - 1)/a_i$; or if $m_i = a_i b_i + 1$ then $a_i^{-1} = m_i - (m_i - 1)/a_i$; or if $m_i = a_i b_i + 1$ then $a_i^{-1} = m_i - (m_i - 1)/a_i$.

We only use an example to expound the usage of “if $m_i = a_i b_i + 1$ then $a_i^{-1} = m_i - (m_i - 1)/a_i$ ”.

Let $a_1 = a_2 = a_3 = a = 18$, $b_1 = 17$, $b_2 = 18$, $b_3 = 19$.

From $m_i = a_i b_i + 1$ we obtain, $m_1 = 18 \times 17 + 1 = 307$, $m_2 = 18 \times 18 + 1 = 325$, $m_3 = 18 \times 19 + 1 = 343$.

$a_1^{-1} = m_1 - (m_1 - 1)/18 = 290$, $a_2^{-1} = m_2 - (m_2 - 1)/18 = 307$, $a_3^{-1} = m_3 - (m_3 - 1)/18 = 324$.

Thus, we obtain the mutually inverse 3-level parameter queues agreeing with the conditions:

(18, 307), (18, 325), (18, 343), (290, 307), (307, 325), (324, 343).

It is not hard to discover that, if $m_i = a_i b_i + 1$ and $1 < c \mid a$, then $a_i^{-1} = m_i - (m_i - 1)/c$. Therefore Skill 2 has many forms of applications.

For example, $9 \mid 18$. When $m_1 = 18 \times 17 + 1 = 307$, $m_2 = 18 \times 18 + 1 = 325$, $m_3 = 18 \times 19 + 1 = 343$, and $a_1 = a_2 = a_3 = 9$, there are $a_1^{-1} = m_1 - (m_1 - 1)/9 = 273$, $a_2^{-1} = m_2 - (m_2 - 1)/9 = 289$, $a_3^{-1} = m_3 - (m_3 - 1)/9 = 305$.

Thus, we obtain mutually inverse 3-level parameter queues agreeing with the conditions:

(9, 307), (9, 325), (9, 343), (273, 307), (289, 325), (305, 343).

Example 2 for XM-1: Use “if $m_i = a_i b_i + 1$ then $a_i^{-1} = m_i - (m_i - 1)/a$ ” to design an XM-1 with the plain text x (i.e., $0 < x < 10^8$) not exceeding decimal numbers of 8 digits.

Based on the following appointments, A side obtains n-level parameter queue agreeing with the conditions (a_1, m_1), ..., (a_n, m_n):

- If $(d_{SJ})_{\text{mod } 51} < 3$ then $n = (d_{SJ})_{\text{mod } 51} + 3$, otherwise $n = (d_{SJ})_{\text{mod } 51}$.
- If $(d_{SJ})_{\text{mod } 714285} < 88000$ then $b = (d_{SJ})_{\text{mod } 714285} + 88000$, otherwise $b = (d_{SJ})_{\text{mod } 714285}$.
- If $(b)_{\text{mod } 2400} < 1137$ then $a = (b)_{\text{mod } 2400} + 1137$, otherwise $a = (b)_{\text{mod } 2400}$.
- $m_1 = ab + 1$, $m_2 = a(b + 5) + 1$, $m_n = a(b + 5(n - 1)) + 1$.
- $a_1 = a_2 = \dots = a_n = a$.

Based on the following appointments, B side obtains n-level parameter queue agreeing with the conditions (a_1^{-1}, m_1), (a_n^{-1}, m_n):

- If $(d_{SJ})_{\text{mod } 51} < 3$ then $n = (d_{SJ})_{\text{mod } 51} + 3$, otherwise $n = (d_{SJ})_{\text{mod } 51}$.
- If $(d_{SJ})_{\text{mod } 714285} < 8800$ then $b = (d_{SJ})_{\text{mod } 714285} + 8800$, otherwise $b = (d_{SJ})_{\text{mod } 714285}$.

- If $(b)_{\text{mod } 2400} < 1137$ then $a = (b)_{\text{mod } 2400} + 1137$, otherwise $a = (b)_{\text{mod } 2400}$.
- $m_1 = ab + 1$, $m_2 = a(b + 5) + 1$, $m_n = a(b + 5(n - 1)) + 1$.
- $a_1^{-1} = m_1 - (m_1 - 1)/a$, $a_2^{-1} = m_2 - (m_2 - 1)/a$, ..., $a_n^{-1} = m_n - (m_n - 1)/a$.

Thus, another XM-1 has been made. The activation process is omitted.

Lemma 2: If $m = b(b + 2)$ then $(b + 1)^2 \equiv 1 \pmod{m}$ or $b + 1 \equiv (b + 1)^{-1} \pmod{m}$.

Proof: $(b + 1)^2 = b^2 + 2b + 1 = b(b + 2) + 1 = m + 1 \equiv 1 \pmod{m}$, and $b + 1 \equiv (b + 1)^{-1} \pmod{m}$. Q.E.D.

Let $a_i = b_i + 1$. From Lemma 2 we obtain:

Skill 3: If $m_i = b_i(b_i + 2)$ then $a_i = a_i^{-1} = b_i + 1$.

Let $b_i = 17 + 2(i - 1)$.

From Skill 3 we have

$m_1 = 17 \times 19 = 323$, $a_1 = a_1^{-1} = 17 + 1 = 18$;

$m_2 = 19 \times 21 = 399$, $a_2 = a_2^{-1} = 19 + 1 = 20$;

$m_3 = 21 \times 23 = 483$, $a_3 = a_3^{-1} = 21 + 1 = 22$.

Hence, we obtain the following mutually inverse 3-level parameter queues agreeing with the conditions:

(18, 323), (20, 399), (22, 483), (18, 323), (20, 399), (22, 483).

Example 3 for XM-1: Use Skill 3 to design an XM-1 with $10^6 < m_1 < m_2 < \dots < m_n < 10^8$, $3 \leq n \leq 50$.

Based on the following appointments A side obtain n-level parameter queue agreeing with the conditions (a_1, m_1), ..., (a_n, m_n):

- if $(d_{SJ})_{\text{mod } 51} < 3$ then $n = (d_{SJ})_{\text{mod } 51} + 3$, otherwise $n = (d_{SJ})_{\text{mod } 51}$.
- If $(d_{SJ})_{\text{mod } 3050} < 1300$ then $b = (d_{SJ})_{\text{mod } 3050} + 1300$, otherwise $b = (d_{SJ})_{\text{mod } 3050}$.
- $m_1 = b(b + 2)$, $m_2 = (b + 2)(b + 4)$, ..., $m_n = (b + 2(n - 1))(b + 2n)$.
- $a_1 = b + 1$, $a_2 = b + 1 + 2$, ..., $a_n = b + 1 + 2(n - 1)$. Or $a_1 = b + 1$, $a_2 = a_1 + 2$, ..., $a_n = a_{n-1} + 2$.

Based on the following appointments B side obtain n-level parameter queue agreeing with the conditions (a_1^{-1}, m_1), (a_n^{-1}, m_n):

- If $(d_{SJ})_{\text{mod } 51} < 3$ then $n = (d_{SJ})_{\text{mod } 51} + 3$, otherwise $n = (d_{SJ})_{\text{mod } 51}$. (That is $2 < n < 51$)
- If $(d_{SJ})_{\text{mod } 3050} < 1300$ then $b = (d_{SJ})_{\text{mod } 3050} + 1300$, otherwise $b = (d_{SJ})_{\text{mod } 3050}$. (That is $1300 \leq b < 3050$)
- $m_1 = b(b + 2)$, $m_2 = (b + 2)(b + 4)$, ..., $m_n = (b + 2(n - 1))(b + 2n)$.
- $a_1^{-1} = b + 1$, $a_2^{-1} = a_1^{-1} + 2$, ..., $a_n^{-1} = a_{n-1}^{-1} + 2$.

Thus, the third XM-1 has been made. (Note: In this example for XM-1 the secret keys of A and B sides are all the same.)

Now, we activate the XM-1 given by Example 3 for XM-1. We still suppose $d_{SJ} = 629629579$.

A side

According to appointment (1), from $(d_{SJ})_{\text{mod } 51} = (629629579)_{\text{mod } 51} = 1 < 3$ we know that, $n = 4$.

According to appointment (2), from $(d_{SJ})_{\text{mod } 3050} = (629629579)_{\text{mod } 3050} = 2085 > 1300$ we know that, $b = 2085$.

According to appointment (3), $m_1 = 2085 \times 2087 = 4351395$, $m_2 = 2087 \times 2089 = 4359743$,

$m_3 = 2089 \times 2091 = 4368099$, $m_4 = 2091 \times 2093 = 4376463$.

According to appointment (4), $a_1 = 2086$, $a_2 = 2088$, $a_3 = 2090$,

$a_4=2092$.

Thus, we obtain 4-level parameter queue agreeing with the conditions:

(2086, 4351395), (2088, 4359743), (2090, 4368099), (2092, 4376463).

B side

According to appointment (1), from

$(d_{SJ})_{\text{mod } 51} = (629629579)_{\text{mod } 51} = 1 < 3$ we know that, $n=4$.

According to appointment (2), from

$(d_{SJ})_{\text{mod } 3050} = (629629579)_{\text{mod } 3050} = 2085 > 1300$ we know that, $b=2085$.

According to appointment (3), $m_1=2085 \times 2087 + 1 = 4351396$, $m_2=2087 \times 2089 + 1 = 4359744$,

$m_3=2089 \times 2091 + 1 = 4368100$, $m_4=2091 \times 2093 + 1 = 4376464$.

According to appointment (4), $a_1^{-1}=2086$, $a_2^{-1}=2088$,

$a_3^{-1}=2090$, $a_4^{-1}=2092$.

Thus, we obtain 4-level parameter queue agreeing with the conditions:

(2086, 4351396), (2088, 4359744), (2090, 4368100), (2092, 4376464).

Lemma 3: If $ab \equiv 1 \pmod{m}$ then

$(a^n)_{\text{mod } m} \cdot (b^n)_{\text{mod } m} \equiv 1 \pmod{m}$.

Proof: From Property 2 we know that, $(a^n)_{\text{mod } m} \equiv a^n \pmod{m}$, $(b^n)_{\text{mod } m} \equiv b^n \pmod{m}$,

Thus, from $ab \equiv 1 \pmod{m}$ we know that,

$(a^n)_{\text{mod } m} \cdot (b^n)_{\text{mod } m} \equiv a^n \cdot b^n \equiv (ab)^n \equiv 1 \pmod{m}$. Q.E.D.

Skill 4: Suppose b_i is an odd number, $m_i = b_i(b_i+2)$. Thus, if $a_i = (4^n)_{\text{mod } m_i}$ then $a_i^{-1} = (((m_i+1)/4)^n)_{\text{mod } m_i}$.

Proof: From b_i is an odd number we know that, $2 \mid b_i+1$, $4 \mid (b_i+1)^2$. From Lemma 2 and $m_i = b_i(b_i+2)$ we know that, $(b_i+1)^2 = m_i+1 \equiv 1 \pmod{m_i}$.

From $4 \mid (b_i+1)^2$ we know that, $4 \mid m_i+1$. From $m_i+1 \equiv 1 \pmod{m_i}$ we know that, $4(m_i+1)/4 \equiv 1 \pmod{m_i}$.

From $4(m_i+1)/4 \equiv 1 \pmod{m_i}$ and Lemma 3 we know that, $(4^n)_{\text{mod } m_i} \cdot (((m_i+1)/4)^n)_{\text{mod } m_i} \equiv 1 \pmod{m_i}$.

Therefore, if $a_i = (4^n)_{\text{mod } m_i}$ then $a_i^{-1} = (((m_i+1)/4)^n)_{\text{mod } m_i}$. Q.E.D.

In Skill 4, let $b_i = 17+2(i-1)$. $m_1 = 17 \times 19 = 323$, $m_2 = 19 \times 21 = 399$, $m_3 = 21 \times 23 = 483$,

Let $a_1 = a_2 = a_3 = 4^2 = 16$. Then, $a_1^{-1} = (((323+1)/4)^2)_{\text{mod } 323} = 101$

$a_2^{-1} = (((399+1)/4)^2)_{\text{mod } 399} = 25$

$a_3^{-1} = (((483+1)/4)^2)_{\text{mod } 483} = 151$

Hence, we obtain the following mutually inverse 3-level parameter queues agreeing with the conditions:

(16, 323), (16, 399), (16, 483), (101, 323), (25, 399), (151, 483).

Example 4 for XM-1: Use Skill 4 to design an XM-1 with $10^6 < m_1 < m_2 \dots < m_n < 10^8$, $3 \leq n \leq 50$.

Based on the following appointments, A side obtains n-level parameter queue agreeing with the conditions $(a_1, m_1), \dots, (a_n, m_n)$:

1. If $(d_{SJ})_{\text{mod } 51} < 3$ then $n = (d_{SJ})_{\text{mod } 51} + 3$, otherwise $n = (d^{SJ})_{\text{mod } 51}$. (That is $2 < n < 51$)
2. If $(d_{SJ})_{\text{mod } 3050} < 1300$ then $b = (d_{SJ})_{\text{mod } 3050} + 1300$, otherwise $b = (d_{SJ})_{\text{mod } 3050}$.
3. If $2 \mid b$ then $b = b+1$. (Note: “ $b=b+1$ ” is an assignment clause)
4. $m_1 = b(b+2)$, $m_2 = (b+2)(b+4)$, $\dots$, $m_n = (b+2(n-1))(b+2n)$.
5. $a_1 = a_2 = \dots = a_n = 4$.

Based on the following appointments, B side obtains n-level parameter queue agreeing with the conditions $(a_1^{-1}, m_1), (a_n^{-1}, m_n)$:

1. If $(d_{SJ})_{\text{mod } 51} < 3$ then $n = (d_{SJ})_{\text{mod } 51} + 3$, otherwise $n = (d^{SJ})_{\text{mod } 51}$. (That is $2 < n < 51$)
2. If $(d^{SJ})_{\text{mod } 3050} < 1300$ then $b = (d_{SJ})_{\text{mod } 3050} + 1300$, otherwise $b = (d_{SJ})_{\text{mod } 3050}$.
3. If $2 \mid b$ then $b = b+1$. (Note: “ $b=b+1$ ” is an assignment clause)
4. $m_1 = b(b+2)$, $m_2 = (b+2)(b+4)$, $\dots$, $m_n = (b+2(n-1))(b+2n)$.
5. $a_1^{-1} = (m_1+1)/4$, $a_2^{-1} = (m_2+1)/4$, $\dots$, $a_n^{-1} = (m_n+1)/4$.

(Note: The first 4 appointments of A side and B side are correspondingly the same.)

Thus, the fourth XM-1 has been made. The activation process is omitted.

The above 4 examples for XM-1 give 4 methods for making XM-1. In every example for XM-1 we use only one skill. Even though, the safety of XM-1 is out of question. In practical applications more mathematical skills can be used to make one XM-1., which will be more colorful.

Besides, we can also use Euclid's second algorithm to find a_1^{-1} , a_n^{-1} . So, there are many methods to make XM-1. The making methods are diversified and randomized, so that the assaulters are unable to assault XM-1.

XM-1 is highly safe, convenient to make, simple in computing method, cheap to run, which means it has many applications.

Elimination of Duplicate Cipher Texts

In the general secret key systems, the same plain text symbol always maps to the same cipher text symbol. The repeat frequency of the plain text is also one of the cipher texts, which means that the statistical attributes of the plain texts are retained in the cipher texts. In order to resolve the risk, it results in, we introduce the “one plain text, more cipher texts” function provided by XM-1.

In I of Theorem 2, when formula (3) satisfies condition “ $x < m_1$, $y_1 < m_2$, $y_{n-1} < m_n$ ”, then by y_n obtained from formula (3) we can obtain x from formula (4). It worth notices that, there is no “ y_n ” in “ $x < m_1$, $y_1 < m_2$, $y_{n-1} < m_n$ ”. That is to say, “ y_n ” is not bound by the conditions.

Carefully examine Theorem 2 we find out that, when formula (4) $y_{n-1} = (a_n^{-1}y_n)_{\text{mod } m_n}$, $\dots$, $x = (a_1^{-1}y_1)_{\text{mod } m_1}$ holds, $y_{n-1} = (a_n^{-1}(y_n + km_n))_{\text{mod } m_n}$, $\dots$, $x = (a_1^{-1}y_1)_{\text{mod } m_1}$ also holds. □□□

The holding of formula (9) means that XM-1 has “one plain text, more cipher texts” function, which is very convenient to eliminate the duplicate cipher texts. When cipher text y_n

occurs the n th time, XM-1 changes y_n into " $y_n + km_n$ ". For every duplicate cipher text to do so, the duplicate cipher texts will be thoroughly removed.

Discussion of practical program 1

In order to demonstrate the functionalities and features of XM-1, we use Python to program the first practical program based on XM-1, called practical program 1. (We will show practical program 1 during the conference.)

1. The programmings of practical program 1 are extremely simple, both the encryption program and the decryption program are no more than 60 Python instructions. They can be ran on the extremely ordinary personal computer.
2. The language of the plain text needing to be encrypted is not restricted. That is to say, the plain text can be any language in the world.
3. In order to verify the encryption/decryption time of the unit plain text (10 digits of decimal numbers), We set the encryption/decryption level n between 3 and 6000000. We find out that for every increase of 210000 of the encryption/decryption level n , the operation time increases 1 second.
4. Practical program 1 has another functionality. Under the circumstances that the encryption program is not revised, when we input the same plain text many times, each time the cipher text obtained is different. And under the circumstances that the decryption program is not revised, the same plain text is obtained. That is to say, under the circumstances that the encryption/decryption system does not revise, we can realize "one time, one cipher text".

Therefore, XM-1 has obvious advantages compared to non-symmetric algorithm RSA and symmetric algorithm AES.

RSA does have some special uses. But, as a dynamic algorithm, it has some shortcomings hard to overcome. Superficially, it is easy for RSA to achieve "one time, one cipher text". Before encryption, so long as B side sends the encryption public key $k_{pub}=(n,e)$ to A side, A side can perform a new encryption. However, as n is a multiplication of two large prime numbers p and q (and $(e, \Phi(n))=1$), each time B side sends a new public key B side must acquire two new large prime numbers. This is a work of great cost.

At the same time, as the cipher text is $y=(x \cdot e) \bmod n$. The encryption process needs to make exponential operation; the consumptions of time and resource are huge.

AES is inferior to XM-1 in the following fields. 1. AES is an encryption/decryption algorithm of at most 3 levels. While XM-1 is a random (or arbitrary) multi-level encryption/decryption

algorithm. Their securities are not in the same magnitude. 2. AES needs a lot of time and resource. While XM-1 needs very little time and resource. 3. AES is hard to achieve "one time, one encryption". While XM-1 can automatically achieve "one time, one encryption".

Acknowledgment

Special notice, the fundamental principle of this algorithm has acquired the invention patent of the People's Republic of China. Patent number: ZL 2019 1 1081775.5

References

1. Christof Paar, Jan Pelzl, Bart Preneel. Understanding Cryptography: A Textbook for Students and Practitioners, Berlin: Springer-Verlag. 2010.
2. Kenneth H Rosen. Elementary Number Theory and Its Applications, Beijing: China Machine Press. 2005.
3. AES Lounge. 2007. <http://www.iaik.tu—graz.ac.at/research/krypto/AES/>.
4. Shekoufeh Neisarian, Elif Bilge Kavun. MT-TMVP: Modular Tiled TMVP-based Polynomial Multiplication for Post-Quantum Cryptography on FPGAs, ePrint Archive. 2025.
5. Tejas Sharma, Ashish Kundu. Security of Operations on Random Numbers: A Review, ePrint Archive. 2025.
6. Linghe Yang, Jian Liu, Jingyi Cui, Guangquan Xu, Yude Bai, et al. Rubato: Provably Post-Quantum Secure and Batched Asynchronous Randomness Beacon, ePrint Archive. 2025.
7. Nicholas Brandt, Miguel Cueto Noval, Christoph U Gunther, Akin Unal, Stella Wahnig. Constrained Verifiable Random Functions Without Obfuscation and Friende, ePrint Archive. 2025. 478-511.
8. Shi Bai, Hansraj Jangir, Elena Kirshanova, Tran Ngo, William Youmans. A Quasi-polynomial Time Algorithm for the Extrapolated Dihedral Coset Problem over Power-of-Two-Moduli, ePrint Archive. 2025. 416-448.
9. Christof Beierle, Phil Hebborn, Gregor Leander, Yevhen Pehuda. Integral Resistance of Block Ciphers with Key Whitening by Modular Addition, ePrint Archive. 2025. 497-529.
10. Fuyuki Kitagawa, Takahiro Matsuda. Adaptive TDF from PKE with Randomness Recoverability and Pseudorandom Ciphertext Property, ePrint Archive. 2025.
11. Klaus Dohmen, Mandy Lange-Geisler. General Multi-Prime Multi-Power RSA: A Generalization of RSA and CRT-RSA to Regular Integers Modulo n , ePrint Archive. 2025.