A Public Key Cryptoscheme Using the Bit-pair Method [*]

Shenghui Su [1, 2], Maozhi Xu [3], Tao Xie [4], and Shuwang Lü [5]

[1] College of Computers, Beijing University of Technology, Beijing 100124
[2] College of Info. Engineering, Yangzhou University, Yangzhou 225009
[3] School of Mathematics, Peking University, Beijing 100871
[4] School of Computers, National University of Defense Technology, Changsha 410073
[5] Graduate School, Chinese Academy of Sciences, Beijing 100039

Abstract: The authors give the definition and property of a bit-pair shadow, and design the three algorithms of a public key cryptoscheme called JUNA which regards a bit-pair as an operation unit, and is based on the multivariate permutation problem (MPP) and the anomalous subset product problem (ASPP). Then, demonstrate that the decryption algorithm is correct, deduce the probability that a plaintext solution is nonunique is nearly zeroth, dissect the time complexities of the algorithms, and analyze the security of the cryptoscheme against extracting a private key from a public key, and recovering a plaintext from a ciphertext by LLL lattice base reduction, adaptive-chosen-ciphertext manner, and meet-in-the-middle dichotomy on the assumption that the IFP, DLP, and SSP of low density can be solved efficiently. Besides, give the conversion from the ASPP to the anomalous subset sum problem (ASSP) through a discrete logarithm. The facts show the bit-pair method increases the density of a related ASSP knapsack with $D > 1$, and decreases the length of modulus of the cryptoscheme with $\lceil \lg M \rceil = 384$, 464, 544, or 640 corresponding to $n = 80$, 96, 112, or 128.

Keywords: Public key cryptoscheme; Anomalous subset sum problem; Bit-pair shadow string; Compact sequence; Lever function

1 Introduction

In [1], we propose a prototypal public key cryptosystem called REESSE1+ which is based on the three new provable problems, contains the five algorithms, and is used for data encryption and digital signing.

In REESSE1+, a ciphertext is defined as $\bar{G} \equiv \prod_{i=1}^{n} C_i^{\underline{b}_i} \,(\%\, M)$, where $\underline{b}_i$ is the bit shadow of a bit b_i [1], and n is the bit-length of a plaintext block.

Let $C_1 \equiv g^{u_1} \,(\%\, M)$, …, $C_n \equiv g^{u_n} \,(\%\, M)$, and $\bar{G} \equiv g^{v} \,(\%\, M)$, where g is a generator of $(\mathbb{Z}_M^*, \cdot)$ which can be found in tolerable subexponential time when the modulus $M < 2^{1024}$ can be factorized [2]. Then solving $\bar{G} \equiv \prod_{i=1}^{n} C_i^{\underline{b}_i} \,(\%\, M)$ for $\underline{b}_1 \ldots \underline{b}_n$ is equivalent to solving

$$\underline{b}_1 u_1 + \ldots + \underline{b}_n u_n \equiv v \,(\%\, \bar{M}). \tag{1}$$

where v may be substituted with $v + k\bar{M}$ along with $k \in [0, n-1]$ [3].

Equation (1) is called an anomalous subset sum problem due to every $\underline{b}_i \in [0, n]$, shortly ASSP [1]. Likewise, due to every $\underline{b}_i \in [0, n]$, $\{u_1, \ldots, u_n\}$ is called a compact sequence [4].

It is not difficult to understand that an ASSP may be converted into a subset sum problem (SSP), and thus the density of an ASSP knapsack is defined as

$$\begin{aligned} D &= \textstyle\sum_{i=1}^{n} \lceil \lg n \rceil / \lceil \lg M \rceil \\ &= n \lceil \lg n \rceil / \lceil \lg M \rceil. \end{aligned} \tag{2}$$

Evidently, the parameters $\lceil \lg M \rceil$ and n have an important influence on the value of D.

In REESSE1+, there are $n = 80$, 96, 112, or 128 and $\lceil \lg M \rceil = 696$, 864, 1030, or 1216. Substituting the parameters with concrete values yields

$D = 80 \times 7 / 696 \approx 0.8046 < 1$ for $n = 80$ and $\lceil \lg M \rceil = 696$,
$D = 96 \times 7 / 864 \approx 0.7778 < 1$ for $n = 96$ and $\lceil \lg M \rceil = 864$,
$D = 112 \times 7 / 1030 \approx 0.7612 < 1$ for $n = 112$ and $\lceil \lg M \rceil = 1030$,
$D = 128 \times 8 / 1216 \approx 0.8421 < 1$ for $n = 128$ and $\lceil \lg M \rceil = 1216$.

The above values mean that the original solution to an ASSP may possibly be found through the LLL lattice basis reduction algorithm [5][6]. However, it is uncertain to find the original solution to the ASSP since $D < 1$ only assure that the shortest vector is unique, and it cannot assure that the vector of the original solution is just the shortest vector or an approximately shortest vector occurring in the final

[*] This work is supported by MOST with Project 2007CB311100 and 2009AA01Z441. Corresponding email: reesse@126.com.

reduced basis.

The LLL algorithm is famous for it has a fatal threat to the classical MH knapsack cryptosystem [7] which produces a ciphertext in the form of a subset sum problem.

To avoid it that the original solution may possibly be found through LLL and to decrease the length of modulus of a cryptoscheme, on the basis of REESSE1+, we propose a new cryptoscheme called JUNA which treats a bit-pair as an operation unit when a bit string is encrypted in this paper.

Throughout the paper, unless otherwise specified, $n \geq 80$ is the bit-length of a plaintext block or the item-length of a sequence, the sign % means "modulo", $\bar{M}$ means "$M-1$" with M prime, lg$x$ denotes the logarithm of x to the base 2, $\neg$ does the opposite value of a bit, $\bar{P}$ does the maximal prime allowed in coprime sequences, $|x|$ does the absolute value of a number x, $\|x\|$ does the order of an element x % M, $|S|$ does the size of a set S, and gcd(a, b) represents the greatest common divisor of two integers. Without ambiguity, "% M" is usually omitted in expressions.

2 Several Definitions

The following definitions lay the stone foundation for the new public key encryption scheme.

2.1 A Coprime Sequence

Definition 1: If $A_1, \ldots, A_n$ are n pairwise distinct positive integers such that $\forall\, A_i, A_j$ ($i \neq j$), either gcd(A_i, A_j) = 1 or gcd(A_i, A_j) = $F \neq 1$ with $(A_i / F) \nmid A_k$ and $(A_j / F) \nmid A_k$ $\forall\, k \neq i, j \in [1, n]$, these integers are called a coprime sequence, denoted by $\{A_1, \ldots, A_n\}$, shortly $\{A_i\}$.

Notice that the elements of a coprime sequence are not necessarily pairwise coprime, but a sequence whose elements are pairwise coprime is a coprime sequence.

Property 1: Let $\{A_1, \ldots, A_n\}$ be a coprime sequence. If randomly select $l \in [1, n]$ elements $A_{x_1}, \ldots, A_{x_l}$ from the sequence, then the mapping from a subset $\{A_{x_1}, \ldots, A_{x_l}\}$ to a subset product $G = \prod_{i=1}^{l} A_{x_i}$ is one-to-one, namely the mapping from $b_1 \ldots b_n$ to $G = \prod_{i=1}^{n} A_i^{b_i}$ is one-to-one, where $b_1 \ldots b_n$ is a bit string.

Refer to [1] for its proof.

2.2 A Bit Shadow

Definition 2: Let $b_1 \ldots b_n \neq 0$ be a bit string. Then $\underline{b}_i$ with $i \in [1, n]$ is called a bit shadow if it comes from such a rule: ① $\underline{b}_i = 0$ if $b_i = 0$, ② $\underline{b}_i = 1$ + the number of successive 0-bits before b_i if $b_i = 1$, or ③ $\underline{b}_i$ = 1 + the number of successive 0-bits before b_i + the number of successive 0-bits after the rightmost 1-bit if b_i is the leftmost 1-bit.

Notice that ③ of this definition is slightly different from that in [1].

For example, let $n = 16$, then when $b_1 \ldots b_{16}$ = 1001000001001100 or 0010010011000100, $\underline{b}_1 \ldots \underline{b}_{16}$ = 3003000006003100 or 0050030031000400.

Fact 1: Let $\underline{b}_1 \ldots \underline{b}_n$ be the bit shadow string of $b_1 \ldots b_n \neq 0$. Then there is $\sum_{i=1}^{n} \underline{b}_i = n$.

Proof:

According to Definition 2, every bit of $b_1 \ldots b_n$ is considered into $\sum_{i=1}^{k} \underline{b}_{x_i}$, where $k \leq n$, and $\underline{b}_{x_1}, \ldots, \underline{b}_{x_k}$ are 1-bit shadows in the string $\underline{b}_1 \ldots \underline{b}_n$, and thus there is $\sum_{i=1}^{k} \underline{b}_{x_i} = n$.

On the other hand, there is $\sum_{j=1}^{n-k} \underline{b}_{y_j} = 0$, where $\underline{b}_{y_1}, \ldots, \underline{b}_{y_{n-k}}$ are 0-bit shadows.

In total, there is $\sum_{i=1}^{n} \underline{b}_i = n$. □

Property 2: Let $\{A_1, \ldots, A_n\}$ be a coprime sequence, and $\underline{b}_1 \ldots \underline{b}_n$ be the bit shadow string of $b_1 \ldots b_n \neq 0$. Then the mapping from $b_1 \ldots b_n$ to $G = \prod_{i=1}^{n} A_i^{\underline{b}_i}$ is one-to-one.

Proof:

Firstly, let $b_1 \ldots b_n$ and $b'_1 \ldots b'_n$ be two different nonzero bit strings, and $\underline{b}_1 \ldots \underline{b}_n$ and $\underline{b}'_1 \ldots \underline{b}'_n$ be the two corresponding bit shadow strings.

If $\underline{b}_1 \ldots \underline{b}_n = \underline{b}'_1 \ldots \underline{b}'_n$, then by Definition 2, there is $b_1 \ldots b_n = b'_1 \ldots b'_n$.

In addition, for any arbitrary bit shadow $\underline{b}_1 \ldots \underline{b}_n$, there always exists a preimage $b_1 \ldots b_n$. Thus, the mapping from $b_1 \ldots b_n$ to $\underline{b}_1 \ldots \underline{b}_n$ is one-to-one.

Secondly, obviously the mapping from $\underline{b}_1 \ldots \underline{b}_n$ to $\prod_{i=1}^{n} A_i^{\underline{b}_i}$ is surjective.

Presuppose that $\prod_{i=1}^{n} A_i^{\underline{b}_i} = \prod_{i=1}^{n} A_i^{\underline{b}'_i}$ for $\underline{b}_1 \ldots \underline{b}_n \neq \underline{b}'_1 \ldots \underline{b}'_n$.

Since $\{A_1, \ldots, A_n\}$ is a coprime sequence, and $A_i^{\underline{b}_i}$ either equals 1 with $\underline{b}_i = 0$ or contains the same prime factors as those of A_i with $\underline{b}_i \neq 0$, we can obtain $\underline{b}_1 \ldots \underline{b}_n = \underline{b}'_1 \ldots \underline{b}'_n$ from $\prod_{i=1}^{n} A_i^{\underline{b}_i} = \prod_{i=1}^{n} A_i^{\underline{b}'_i}$, which is in direct contradiction to $\underline{b}_1 \ldots \underline{b}_n \neq \underline{b}'_1 \ldots \underline{b}'_n$.

Therefore, the mapping from $\underline{b}_1 \ldots \underline{b}_n$ to $\prod_{i=1}^{n} A_i^{\underline{b}_i}$ is injective [8].

In summary, the mapping from $\underline{b}_1 \ldots \underline{b}_n$ to $\prod_{i=1}^{n} A_i^{\underline{b}_i}$ is one-to-one, and further the mapping from $b_1 \ldots b_n$ to $\prod_{i=1}^{n} A_i^{\underline{b}_i}$ is also one-to-one. □

2.3 A Bit-pair Shadow

It is well understood that a public key cryptosystem is mainly used for transmitting a symmetric key. Assume that $b_1 \ldots b_n$ is a symmetric key. At present, to prevent exhaustive search, namely brute force attack, n should be no less than 80 [9].

To make the modulus M of the new cryptoscheme comparatively small, we will utilize the idea of a bit-pair string with 2 to 3.

In this way, the length of a coprime sequence is changed to $3n/2$, namely $\{A_1, \ldots, A_n\}$ is substituted with $\{A_1, A_2, A_3, \ldots, A_{3n/2-2}, A_{3n/2-1}, A_{3n/2}\}$ that may be orderly partitioned into $n/2$ triples of which each comprises 3 elements: A_{3i-2}, A_{3i-1}, A_{3i} with $i \in [1, n/2]$. Likewise, a non-coprime sequence $\{C_1, \ldots, C_n\}$ is substituted with $\{C_1, C_2, C_3, \ldots, C_{3n/2-2}, C_{3n/2-1}, C_{3n/2}\}$, where C_i with $i \in [1, 3n/2]$ is acquired from A_i and other private parameters.

***Definition 3**:* Let $\{A_1, \ldots, A_{3n/2}\}$ be a coprime sequence. Orderly partition a bit string $b_1 \ldots b_n$ into $n/2$ pairs $B_1, \ldots, B_{n/2}$, where B_i with $i \in [1, n/2]$ has four state: 00, 01, 10, and 11 which correspond to 1, A_{3i-2}, A_{3i-1}, and A_{3i} respectively. Then $B_1, \ldots, B_{n/2}$ is called a bit-pair string, shortly $B_1 \ldots B_{n/2}$.

***Property 3**:* Let $\{A_1, \ldots, A_{3n/2}\}$ be a coprime sequence, and $B_1 \ldots B_{n/2}$ be a nonzero bit-pair string. Then the mapping from $B_1 \ldots B_{n/2}$ to $G' = \prod_{i=1}^{n/2} (A_{3(i-1)+B_i})^{\lceil B_i/3 \rceil}$ with $A_0 = 1$ is one-to-one, where $\lceil B_i / 3 \rceil = 0$ or 1, and G' is called a coprime subsequence product.

Its proof is parallel to that of Property 1 in [1].

***Definition 4**:* Let $B_1 \ldots B_{n/2}$ be a nonzero bit-pair string. Then $\underline{B}_i$ with $i \in [1, n/2]$ is called a bit-pair shadow if it comes from such a rule: ① $\underline{B}_i = 0$ if $B_i = 00$, ② $\underline{B}_i = 1 +$ the number of successive 00-pairs before B_i if $B_i \neq 00$, or ③ $\underline{B}_i = 1 +$ the number of successive 00-pairs before B_i + the number of successive 00-pairs after the rightmost non-00-pair if B_i is the leftmost non-00-pair.

For example, let $n = 16$, then when $B_1 \ldots B_8 = 1001000001001100$ or 0010010011000100, $\underline{B}_1 \ldots \underline{B}_8 = 21003020$ or 03102020.

Fact 2: Let $\underline{B}_1 \ldots \underline{B}_{n/2}$ be the bit-pair shadow string of $B_1 \ldots B_{n/2} \neq 0$. Then there is $\sum_{i=1}^{n/2} \underline{B}_i = n/2$.

Proof:

According to Definition 4, every pair of $B_1 \ldots B_{n/2}$ is considered into $\sum_{i=1}^{k} \underline{B}_{x_i}$, where $k \leq n/2$, and $\underline{B}_{x_1}, \ldots, \underline{B}_{x_k}$ are non-00-pair shadows in the string $\underline{B}_1 \ldots \underline{B}_{n/2}$, and thus there is $\sum_{i=1}^{k} \underline{B}_{x_i} = n/2$.

On the other hand, there is $\sum_{j=1}^{n/2-k} \underline{B}_{y_j} = 0$, where $\underline{B}_{y_1}, \ldots, \underline{B}_{y_{n/2-k}}$ are 00-pair shadows.

In total, there is $\sum_{i=1}^{n/2} \underline{B}_i = n/2$. □

***Property 4**:* Let $\{A_1, \ldots, A_{3n/2}\}$ be a coprime sequence, and $\underline{B}_1 \ldots \underline{B}_{n/2}$ be the bit-pair shadow string of $B_1 \ldots B_{n/2} \neq 0$. Then the mapping from $B_1 \ldots B_{n/2}$ to $G = \prod_{i=1}^{n/2} (A_{3(i-1)+B_i})^{\underline{B}_i}$ with $A_0 = 1$ is one-to-one, where G is called an anomalous coprime subsequence product.

Its proof is parallel to that of Property 2 in Section 2.2.

Property 3 and 4 manifest that G' or G may still act as a trapdoor component under bit-pair string circumstances.

2.4 A Lever Function

Considering a bit-pair string, in the following text, let $\tilde{n} = 3n/2$, where $n \leq 128$.

***Definition 5**:* The secret parameter $\ell(i)$ in the key transform of a public key cryptoscheme is called a lever function, if it has the following features:

- $\ell(.)$ is an injection from the domain $\{1, \ldots, \tilde{n}\}$ to the codomain $\Omega \subset \{5, \ldots, \bar{M}\}$, where $\bar{M}$ is large;
- the mapping between i and $\ell(i)$ is established randomly without an analytical expression;
- an attacker has to be faced with all the arrangements of n elements in Ω when extracting a related private key from a public key;
- the owner of a private key only needs to consider the accumulative sum of n elements in Ω when recovering a related plaintext from a ciphertext.

The latter two points manifest that if n is large enough, it is infeasible for the attacker to search all the permutations of elements in Ω exhaustively while the decryption of a normal ciphertext is feasible in some time being polynomial in n. Thus, there are the large amount of calculation on $\ell(.)$ at "a public terminal", and the small amount of calculation on $\ell(.)$ at "a private terminal".

Notice that ① in modular $\bar{M}$ arithmetic, $-x$ represents $\bar{M} - x$; ② the number of elements of Ω is not less than n; ③ considering the speed of decryption, the absolute values of all the elements should be comparatively small; ④ the lower limit 5 will make seeking the root W from $W^{\ell(i)} \equiv A_i^{-1} C_i$ (% M) face an unsolvable Galois group when $A_i \le 1201$ is guessed [10].

Concretely to the JUNA cryptoscheme, $\ell(i)$ in the key transform $C_i \equiv (A_i W^{\ell(i)})^{\delta}$ (% M) with $i \in [1, \tilde{n}]$ is an exponent.

Property 5 **(Indeterminacy of $\ell(.)$):** Let $\delta = 1$ and $C_i \equiv (A_i W^{\ell(i)})^{\delta}$ (% M) with $\ell(i) \in \Omega = \{5, 6, \ldots, \tilde{n} + 4\}$ and $A_i \in \Lambda = \{2, 3, \ldots, \text{Ꝑ} \mid \text{Ꝑ} \le 1201\}$ for $i = 1, \ldots, \tilde{n}$. Then $\forall\, W\, (\|W\| \ne \bar{M}) \in (1, \bar{M})$ and $\forall\, x, y, z\ (x \ne y \ne z) \in [1, \tilde{n}]$,

① when $\ell(x) + \ell(y) = \ell(z)$, there is $\ell(x) + \|W\| + \ell(y) + \|W\| \ne \ell(z) + \|W\|$ (% $\bar{M}$);

② when $\ell(x) + \ell(y) \ne \ell(z)$, there always exist

$$C_x \equiv A'_x W'^{\ell'(x)} \text{ (\% } M\text{)},\ C_y \equiv A'_y W'^{\ell'(y)} \text{ (\% } M\text{)}, \text{ and } C_z \equiv A'_z W'^{\ell'(z)} \text{ (\% } M\text{)}$$

such that $\ell'(x) + \ell'(y) \equiv \ell'(z)$ (% $\bar{M}$) with $A'_z \le \text{Ꝑ}$.

Refer to [1] for its proof.

Notice that according to the proof of Property 5 in [1], it is not difficult to understand that when $\Omega = \{5, 6, \ldots, \tilde{n} + 4\}$ is substituted with $\Omega = \{+/-5, +/-7, \ldots, +/-(2\tilde{n} + 3)\}$, where "+/–" means the selection of the "+" sign or the "–" sign, Property 5 still holds.

3 Design of the JUNA Cryptoscheme

In this new scheme, two adjacent bits are treated as a unit, namely a bit-pair string $B_1 \ldots B_{n/2}$ is used to represent a related plaintext block $b_1 \ldots b_n \ne 0$.

3.1 The Key Generation Algorithm

Let $p_1, \ldots, p_{\tilde{n}}$ be the first $\tilde{n}$ primes in the natural set which may constitute a smallest coprime sequence.

Considering decryption swiftness, the absolute values of elements of Ω should be as small as possible, and every three successive elements of Ω should be in a triple conforming with 2 bits to 3 items.

Let $\Lambda = \{2, \ldots, \text{Ꝑ}\}$, where $\text{Ꝑ} = 863, 937, 991$, or 1201 as $n = 80, 96, 112$, or 128.

Assume that $\bar{A}_i$ is the maximum in $(A_{3i-2}, A_{3i-1}, A_{3i})$ for $i = 1, \ldots, n/2$.

The following algorithm is generally employed by the owner of a key pair.

```
Input: the integer $n$; the set $\Lambda$; the first $\tilde{n}$ primes in the natural set.
S1: Randomly generate $\Omega =$
    $\{(+/-(6j-1), +/-(6j+1), +/-(6j+3))_{\mathrm{P}} \mid j = 1, \ldots, n/2\}$.
S2: Produce an odd coprime sequence $\{A_1, \ldots, A_{\tilde{n}} \mid A_i \in \Lambda\}$.
    Arrange $\bar{A}_1, \ldots, \bar{A}_{n/2}$ in descending order into $\bar{A}_{x_1}, \ldots, \bar{A}_{x_{n/2}}$.
S3: Find a prime $M > \bar{A}_{x_1}^{\,n/4+1} \prod_{i=2}^{n/4} \bar{A}_{x_i}$ making $\prod_{i=1}^{k} p_i^{e_i} \mid \bar{M}$,
    where $k$ meets $\prod_{i=1}^{k} e_i \ge 2^{10}$ and $p_k < \tilde{n}/2$.
S4: Generate pairwise distinct $(\ell(3i-2), \ell(3i-1), \ell(3i)) \in \Omega$
    for $i = 1, \ldots, n/2$.
S5: Randomly pick $W \in (1, \bar{M})$ making $\|W\| \ge 2^{n-20}$.
    Randomly pick $\delta \in (1, \bar{M})$ making $\gcd(\delta, \bar{M}) = 1$.
S6: Compute $C_i \leftarrow (A_i W^{\ell(i)})^{\delta}$ % $M$ for $i = 1, \ldots, \tilde{n}$.
Output: a public key $(\{C_i\}, M)$; a private key $(\{A_i\}, W, \delta, M)$. $\{(\ell(3i-2), \ell(3i-1), \ell(3i))\}$ is discarded.
```

Notice that

① at S1, $\Omega = \{(+/-(6j-1), +/-(6j+1), +/-(6j+3))_{\mathrm{P}} \mid j = 1, \ldots, n/2\}$ indicates that Ω is one of $(3!)^{n/2} 2^{\tilde{n}}$ potential sets consisting of 3-tuple elements, where "+/–" means the selection of the "+" sign or the "–" sign, and the subscript P means that $(+/-(6j-1), +/-(6j+1), +/-(6j+3))_{\mathrm{P}}$ is a permutation of $(+/-(6j-1), +/-(6j+1), +/-(6j+3))$.

② at S2, $\gcd(A_{3i-2}, A_{3i-1}, A_{3i}) \neq 1$ ($i \in [1, n/2]$) is allowed — $(3^3, 3^2, 3)$ for example since only one of three elements will occur in G;

③ at S3, the inequation $M > \bar{A}_{x_1}{}^{n/4+1} \prod_{i=2}^{n/4} \bar{A}_{x_i}$ assures that when n = 80, 96, 112, or 128, there exists $\lceil \lg M \rceil$ = 384, 464, 544, or 640;

④ at S5, let $W \equiv g^{\bar{M}/F} (\% M)$, then $\|W\| = \bar{M} / \gcd(\bar{M}, \bar{M}/F)$ [10], where $F \geq 2^{n-20}$ is a factor of $\bar{M}$, and g is a generator by Algorithm 4.80 in Section 4.6 of [11].

Definition 6: Given ($\{C_i\}$, M), seeking the original ($\{A_i\}$, $\{\ell(i)\}$, W, δ) from $C_i = (A_i W^{\ell(i)})^{\delta} \% M$ with $A_i \in \Lambda = \{2, \ldots, Ᵽ\}$ and $\ell(i)$ from $\Omega = \{(+/-(6j-1), +/-(6j+1), +/-(6j+3))_P \mid j = 1, \ldots, n/2\}$ for $i = 1, \ldots, \tilde{n}$ is referred to as a multivariate permutation problem (MPP) [1].

3.2 The Encryption Algorithm

This algorithm is employed by a person who wants to encrypt plaintexts.

Input: a public key ($\{C_1, \ldots, C_{\tilde{n}}\}$, M); the bit-pair string $B_1 \ldots B_{n/2}$ of a plaintext block $b_1 \ldots b_n \neq 0$.

Notice that if the number of 00-pairs in $B_1 \ldots B_{n/2}$ is larger than $n/4$, let $b_1 \ldots b_n = \neg b_1 \ldots \neg b_n$ in order that a related ciphertext can be decrypted correctly according to the constraint on M.

```
S1: Set $C_0 \leftarrow 1$, $k \leftarrow 0$, $i \leftarrow 1$, $s \leftarrow 0$.
S2: If $B_i = 00$ then
        let $k \leftarrow k + 1$, $\underline{B}_i \leftarrow 0$
    else
        let $\underline{B}_i \leftarrow k + 1$, $k \leftarrow 0$;
        if $s \neq 0$ then $s \leftarrow i$.
S3: Let $i \leftarrow i + 1$.
    If $i \leq n / 2$ then goto S2.
S4: If $k \neq 0$ then let $\underline{B}_s \leftarrow \underline{B}_s + k$.
S5: Compute $\bar{G} \leftarrow \prod_{i=1}^{n/2} (C_{3(i-1)+B_i})^{\underline{B}_i} \% M$.
Output: the ciphertext $\bar{G}$.
```

We see that a JUNA ciphertext $\bar{G} \equiv \prod_{i=1}^{n/2} (C_{3(i-1)+B_i})^{\underline{B}_i}$ ($\% M$) is different from an Naccache-Stern ciphertext $c \equiv \prod_{i=1}^{n} v_i^{b_i}$ ($\% M$) [12], where $v_i \equiv p_i^{1/\delta}$ ($\% M$) is a public key.

Definition 7: Given ($\{C_1, \ldots, C_{3n/2}\}$, M) and $\bar{G}$, seeking $B_1 \ldots B_{n/2}$ from $\bar{G}_1 \equiv \prod_{i=1}^{n/2} (C_{3(i-1)+B_i})^{\lceil \underline{B}_i / 3 \rceil}$ ($\% M$) with $C_0 = 1$ is referred to as a subset product problem (SPP), where $B_1 \ldots B_{n/2}$ is the bit-pair string of $b_1 \ldots b_n \neq 0$ [1].

Definition 8: Given ($\{C_1, \ldots, C_{3n/2}\}$, M) and $\bar{G}$, seeking $\underline{B}_1 \ldots \underline{B}_{n/2}$ from $\bar{G} \equiv \prod_{i=1}^{n/2} (C_{3(i-1)+B_i})^{\underline{B}_i}$ ($\% M$) with $C_0 = 1$ is referred to as an anomalous subset product problem (ASPP), where $\underline{B}_1 \ldots \underline{B}_{n/2}$ is the bit-pair shadow string of $B_1 \ldots B_{n/2}$, and $B_1 \ldots B_{n/2}$ is the bit-pair string of $b_1 \ldots b_n \neq 0$ [1].

3.3 The Decryption Algorithm

This algorithm is employed by a person who wants to decrypt ciphertexts.

Input: a private key ($\{A_1, \ldots, A_{\tilde{n}}\}$, W, δ, M); a ciphertext $\bar{G}$.

Notice that due to $\sum_{i=1}^{n/2} \underline{B}_i = n/2$ and $\ell(3(i-1)+B_i)$ odd (excluding $\ell(0) = 0$), $\underline{k} = \sum_{i=1}^{n/2} \underline{B}_i \ell(3(i-1)+B_i)$ must be even.

```
S1: Compute $Z_0 \leftarrow \bar{G}^{\delta^{-1}} \% M$.
    Set $Z_1 \leftarrow Z_0$, $h \leftarrow 0$.
S2: If $2 \mid Z_h$ then do $Z_h \leftarrow Z_h W^{2(-1)^h} \% M$, goto S2.
S3: Set $B_1 \ldots B_{n/2} \leftarrow 0$, $j \leftarrow 0$, $k \leftarrow 0$, $i \leftarrow 1$, $s \leftarrow 0$, $G \leftarrow Z_h$.
S4: If $A_{3i-j}{}^{k+1} \mid G$ then
        do $G \leftarrow G / A_{3i-j}{}^{k+1}$, $B_i \leftarrow 3 - j$, $k \leftarrow 0$;
        if $s \neq 0$ then $s \leftarrow 3i - j$ else null
    else
        let $j \leftarrow j + 1$;
        if $j \leq 2$ then goto S4 else let $k \leftarrow k + 1$.
S5: Let $i \leftarrow i + 1$.
    If $i \leq n / 2$ and $G \neq 1$ then set $j \leftarrow 0$, goto S4.
```

S6: If $k \ne 0$ and $A_s^{\,k} \,|\, G$ then do $G \leftarrow G \,/\, A_s^{\,k}$.
S7: If $G \ne 1$ then
set $h \leftarrow \neg h$, do $Z_h \leftarrow Z_h W^{2(-1)^h}$ % M, goto S2.
Output: the original plaintext block $B_1 \ldots B_{n/2}$, namely $b_1 \ldots b_n$.
Notice that as long as $\bar{G}$ is a true ciphertext, the algorithm can always terminate normally.

4 Correctness, Uniqueness, and Complexity

In this section, we discuss whether a ciphertext can be decrypted correctly, a plaintext solution is unique, and a decryption process can be finished in polynomial time.

4.1 Correctness of the Decryption Algorithm

Because $(\mathbb{Z}_M^*, \cdot)$ is an Abelian group, namely a commutative group, $\forall \underline{k} \in [1, \bar{M}]$, there is

$$W^{\underline{k}}(W^{-1})^{\underline{k}} \equiv W^{\underline{k}}(W^{\underline{k}})^{-1} \equiv 1 \ (\%\ M),$$

where $W \in [1, \bar{M}]$ is any arbitrary integer.

Fact 3*:* Let $\underline{k} = \sum_{i=1}^{n/2} \underline{B}_i \ell(3(i-1)+B_i)$ % $\bar{M}$ with $\ell(0) = 0$, where $\underline{B}_1 \ldots \underline{B}_{n/2}$ is the bit-pair shadow string of $B_1 \ldots B_{n/2}$ corresponding to $b_1 \ldots b_n \ne 0$. Then $\bar{G}^{\delta^{-1}}(W^{-1})^{\underline{k}} \equiv \prod_{i=1}^{n/2} (A_{3(i-1)+B_i})^{\underline{B}_i}$ (% M).

Proof:

Let $b_1 \ldots b_n$, namely $B_1 \ldots B_{n/2}$ be an n-bit plaintext block.

Additionally, let $A_0 = 1$.

According to the key generator, the encryption algorithm, and $\sum_{i=1}^{n/2} \underline{B}_i = n/2$, there is

$$\begin{aligned}\bar{G} &\equiv \prod_{i=1}^{n/2} (C_{3(i-1)+B_i})^{\underline{B}_i} \\ &\equiv \prod_{i=1}^{n/2} ((A_{3(i-1)+B_i} W^{\ell(3(i-1)+B_i)})^{\delta})^{\underline{B}_i} \\ &\equiv W^{(\sum_{i=1}^{n/2} \underline{B}_i \ell(3(i-1)+B_i))\delta} \prod_{i=1}^{n/2} (A_{3(i-1)+B_i})^{\delta \underline{B}_i} \\ &\equiv W^{\underline{k}\delta} \prod_{i=1}^{n/2} (A_{3(i-1)+B_i})^{\delta \underline{B}_i} \ (\%\ M).\end{aligned}$$

Further, raising either side of the above congruence to the δ^{-1}-th yields

$$\begin{aligned}\bar{G}^{\delta^{-1}} &\equiv (W^{\underline{k}\delta} \prod_{i=1}^{n/2} (A_{3(i-1)+B_i})^{\delta \underline{B}_i})^{\delta^{-1}} \\ &\equiv W^{\underline{k}} \prod_{i=1}^{n/2} (A_{3(i-1)+B_i})^{\underline{B}_i} \ (\%\ M).\end{aligned}$$

Multiplying either side of the just above congruence by $(W^{-1})^{\underline{k}}$ yields

$$\begin{aligned}\bar{G}^{\delta^{-1}}(W^{-1})^{\underline{k}} &\equiv W^{\underline{k}} \prod_{i=1}^{n/2} (A_{3(i-1)+B_i})^{\underline{B}_i} (W^{-1})^{\underline{k}} \\ &\equiv \prod_{i=1}^{n/2} (A_{3(i-1)+B_i})^{\underline{B}_i} \\ &\equiv G \ (\%\ M).\end{aligned}$$

Clearly, the above process also gives a method of seeking G at one time. □

Notice that in practice, $\underline{k}$ is unknowable in advance.

However, because $|\underline{k}| < n(2\tilde{n} + 3)/2 = 3n(n+1)/2$ is comparatively small, we may search $\underline{k}$ heuristically by multiplying W^{-2} or W^2 and verifying whether $G = 1$ after it is divided exactly by some $A_{3i-j}^{\;k+1}$. It is known from the decryption algorithm that the original $B_1 \ldots B_{n/2}$ will be acquired at the same time the condition $G = 1$ is satisfied.

4.2 Uniqueness of a Plaintext Solution

Because $\{C_1, \ldots, C_{\tilde{n}}\}$ is a non-coprime sequence, the mapping from $B_1 \ldots B_{n/2}$ to $\prod_{i=1}^{n/2} (C_{3(i-1)+B_i})^{\underline{B}_i}$ % $M = \bar{G}$ is theoretically many-to-one. It might possibly result in the nonuniqueness of a plaintext solution $B_1 \ldots B_{n/2}$ when $\bar{G}$ is being unveiled.

Suppose that a ciphertext $\bar{G}$ can be obtained respectively from two different bit-pair strings $B_1 \ldots B_{n/2}$ and $B'_1 \ldots B'_{n/2}$. Then,

$$\bar{G} \equiv \prod_{i=1}^{n/2} (C_{3(i-1)+B_i})^{\underline{B}_i} \equiv \prod_{i=1}^{n/2} (C_{3(i-1)+B'_i})^{\underline{B}'_i} \ (\%\ M).$$

That is,

$$\prod_{i=1}^{n/2} (A_{3(i-1)+B_i} W^{\ell(3(i-1)+B_i)})^{\delta \underline{B}_i} \equiv \prod_{i=1}^{n/2} (A_{3(i-1)+B'_i} W^{\ell(3(i-1)+B'_i)})^{\delta \underline{B}'_i} \ (\%\ M).$$

Further, owing to $\sum_{i=1}^{n/2} \underline{B}_i = \sum_{i=1}^{n/2} \underline{B}'_i = n/2$, there is

$$W^{\underline{k}\delta} \prod_{i=1}^{n/2} (A_{3(i-1)+B_i})^{\delta \underline{B}_i} \equiv W^{\underline{k}'\delta} \prod_{i=1}^{n/2} (A_{3(i-1)+B'_i})^{\delta \underline{B}'_i} \ (\%\ M),$$

where $\underline{k} = \sum_{i=1}^{n/2} \underline{B}_i \ell(3(i-1)+B_i)$, and $\underline{k}' = \sum_{i=1}^{n/2} \underline{B}'_i \ell(3(i-1)+B'_i)$ % $\bar{M}$ with $\ell(0) = 0$.

Raising either side of the above congruence to the δ^{-1}-th power yields

$$W^{\underline{k}} \prod_{i=1}^{n/2} (A_{3(i-1)+B_i})^{\underline{B}_i} \equiv W^{\underline{k}'} \prod_{i=1}^{n/2} (A_{3(i-1)+B'_i})^{\underline{B}'_i} \ (\%\ M).$$

Without loss of generality, let $\underline{k} \ge \underline{k}'$. Because $(\mathbb{Z}_M^*, \cdot)$ is an Abelian group, there is

$$W^{\underline{k}-\underline{k}'} \equiv \prod_{i=1}^{n/2} (A_{3(i-1)+B'_i})^{\underline{B}'_i} (\prod_{i=1}^{n/2} (A_{3(i-1)+B_i})^{\underline{B}_i})^{-1} \ (\%\ M).$$

Let $\theta \equiv \prod_{i=1}^{n/2} (A_{3(i-1)+B'_i})^{\underline{B}'_i} (\prod_{i=1}^{n/2} (A_{3(i-1)+B_i})^{\underline{B}_i})^{-1}$ (% M), namely $\theta \equiv W^{\underline{k}-\underline{k}'}$ (% M).

This congruence signifies when the plaintext $B_1 \ldots B_{n/2}$ is not unique, the value of W must be relevant to θ. The contrapositive assertion equivalent to it is that if the value of W is irrelevant to θ, $B_1 \ldots B_{n/2}$ will be unique. Thus, we need to consider the probability that W takes a value relevant to θ.

If an adversary tries to attack an 80-bit symmetric key through the exhaustive search, and a computer can verify trillion values per second, it will take 38334 years for the adversary to verify all the potential values. Hence, currently 80 bits are quite enough for the security of a symmetric key.

$B_1 \ldots B_{n/2}$ contains n bits which indicates $\prod_{i=1}^{n/2} (A_{3(i-1)+B_i})^{\underline{B}_i}$ has 2^n potential values, and thus the number of potential values of θ is at most $2^n \times 2^n$. Notice that because $A_1^{-1}, \ldots, A_{\tilde{n}}^{-1}$ are not necessarily coprime, some values of θ may possibly occur repeatedly.

Because $|\underline{k} - \underline{k}'| < 3n(n+1) \le 47601 \approx 2^{16}$ as $n \le 128$, and W has at most 2^{16} solutions to every θ, the probability that W takes a value relevant to θ is at most $2^{16}2^{2n}/M$.

When $n \ge 80$, there is $2^{16}2^{2n}/M \le 2^{176}/2^{384} = 1/2^{208}$ (notice that when n = 80, 96, 112, or 128, there is $\lceil \lg M \rceil$ = 384, 464, 544, or 640), which is close to zero. The probability will further decrease when W is a prime since the solutions to θ lean to being composite integers in the average case.

In addition, if you please, resorting to $\sum_{i=1}^{n/2} \underline{B}_i = n/2$, may exclude some unoriginal plaintext solutions.

4.3 Time Complexities of the Algorithms

The running time, namely time complexity of an algorithm on an input is measured in the number of bit operations [11], and it has an asymptotic implication. According to [11], a modular addition will take $O(\lg M)$ bit operations, and a modular multiplication will take $O(2\lg^2 M)$ bit operations.

4.3.1 Running Time of the Key Generator

In the key generator, the steps which exert dominant effects on running time are S4 and S5.

At S4, seeking W will take $O(6(\ln\ln \bar{M})\lg^3 M)$. For every i, S5 contains two modular powers $W^{\ell(i)}$ and $(A_i W^{\ell(i)})^{\delta}$ of which the former is equivalent to $4\lg \tilde{n}$ modular multiplications, and not dominant. Thus, the running time of the key generator is $O((\ln\ln \bar{M} + \tilde{n})\lg^3 M)$.

4.3.2 Running Time of the Encryption Algorithm

In the encryption algorithm, the dominant step is S5.

Due to $\sum_{i=1}^{n/2} \underline{B}_i = n/2$, S5 contains $(n / 2 - 1)$ modular multiplications. Thus, the running time of the encryption algorithm is $O(n\lg^2 M)$.

4.3.3 Running Time of the Decryption Algorithm

In the decryption algorithm, the dominant step is S2 which pairs with S7 to construct a loop.

It is easy to see that the number of times of executing the loop S2 $\leftrightarrow$ S7 which mainly contain a modular multiplication is $\underline{k} = \sum_{i=1}^{n/2} \underline{B}_i \ell(3(i-1)+B_i)$, where $\underline{B}_i$ is relevant to a plaintext block, and $\ell(3(i-1)+B_i)$ is relevant to the set Ω which is indeterminate. Therefore, it is very difficult accurately to know the value of $\underline{k}$.

When a plaintext block $B_1 \ldots B_{n/2}$ contains $n/4$ successive 00-pairs, we may obtain the maximal value of $\underline{k}$, where $\underline{k}$ is considered only as a positive number.

The maximal value of $\underline{k}$ is

$$\begin{aligned} \underline{k} &= (n / 4 + 1)(2\tilde{n} + 3) + (2\tilde{n} - 3) + (2\tilde{n} - 9) + \ldots + (2\tilde{n} + 3 - 6(n / 4 - 1)) \\ &= (3 / 4)(n + 4)(n + 1) + (3 / 16)(3n + 4)(n - 4) \\ &= (3 / 16)n(7n + 12). \end{aligned}$$

Similarly, when a plaintext block $B_1 \ldots B_{n/2}$ contains suitable bit-pairs, we may obtain the minimal value of $\underline{k}$ which equals 0.

In summary, the simply expected value of $\underline{k}$ is $(3 / 32)n(7n + 12) \approx (21 / 32)n^2$.

Again considering that $\underline{k}$ is possibly negative, and $W^{2(-1)^h}$ is multiplied every time, the simply expected value of $\underline{k}$ should be $2(1 / 2)(21 / 32)n^2$ which is still $(21 / 32)n^2$.

Thus, the simply expected running time of the decryption algorithm is $O((21 / 32)n^2\lg^2 M)$.

However, in practice, because 2^n $\underline{k}$-values will distribute at $(3 / 32)n(7n + 12)$ integral points 0, 2, 4, …, $(3 / 16)n(7n + 12)$ which is the maximal possible range, and the probability that $\underline{k}$ takes large integers is comparatively small, the concrete running time of a decryption process will be far smaller than $O((21 / 32)n^2\lg^2 M)$.

5 Analysis of Security of a JUNA Private Key

In this section, we will analyze the security of the new cryptoscheme against extracting a related private key from a public key.

In cryptanalysis, we suppose that the integer factorization problem (IFP) $N = pq$ with $\lceil \lg N \rceil < 1024$ [2], the discrete logarithm problem (DLP) $y \equiv g^x$ (% M) with $\lceil \lg M \rceil < 1024$ [13][14], and the subset sum problem of low density (SSP) $s \equiv \sum_{i=1}^{n} C_i b_i$ (% M) with $D \approx n / \lceil \lg M \rceil < 1$ and $n < 1024$ [7] can be solved in tolerable subexponential time or in polynomial time [15].

5.1 A Property of the MPP

In the new cryptoscheme, there is the MPP $C_i = (A_i W^{\ell(i)})^\delta$ % M with $A_i \in \Lambda$ and $\ell(i)$ from $\Omega = \{(+/-(6j-1), +/-(6j+1), +/-(6j+3))_P \mid j = 1, …, n/2\}$ for $i = 1, …, \tilde{n}$. Notice that the structure of the set Ω consisting of triples has no change in essence compared with that Ω in [1].

According to Definition 6, the MPP has the following property.

Property 6: The MPP $C_i = (A_i W^{\ell(i)})^\delta$ % M with $A_i \in \Lambda = \{2, …, \mathcal{P}\}$ and $\ell(i)$ from $\Omega = \{(+/-(6j-1), +/-(6j+1), +/-(6j+3))_P \mid j = 1, …, n/2\}$ for $i = 1, …, \tilde{n}$ is computationally at least equivalent to the DLP in the same prime field.

Refer to Section 4.1 of [1] for its proof.

5.2 Attack by Interaction of the Key Transform Items

In the key transform $C_i \equiv (A_i W^{\ell(i)})^\delta$ (% M), the parameters $A_i \in \Lambda = \{2, 3, …, \mathcal{P} \mid \mathcal{P} = 863, 937, 991, \text{or } 1201\}$ and $\ell(i)$ from $\Omega = \{(+/-(6j-1), +/-(6j+1), +/-(6j+3))_P \mid j = 1, …, n/2\}$ are vulnerable.

5.2.1 Eliminating *W* through $\ell(x_1) + \ell(x_2) = \ell(y_1) + \ell(y_2)$

$\forall\, x_1, x_2, y_1, y_2 \in [1, n]$, assume that $\ell(x_1) + \ell(x_2) = \ell(y_1) + \ell(y_2)$.

Let $G_z \equiv C_{x_1} C_{x_2} (C_{y_1} C_{y_2})^{-1}$ (% M), namely

$$G_z \equiv (A_{x_1} A_{x_2} (A_{y_1} A_{y_2})^{-1})^\delta \text{ (\% } M).$$

If an adversary divines the values of $A_{x_1}, A_{x_2}, A_{y_1}, A_{y_2} \in \Lambda$, he may compute δ through a discrete logarithm in $L_M[1/3, 1.923]$ time, where $M < 2^{640}$.

However, a concrete Ω is one of $(2^3 3!)^{n/2}$ potential sets consisting of triples, indeterminate, and unknown due to $|\Omega| = n / 2$ and $\Omega = \{(+/-(6j-1), +/-(6j+1), +/-(6j+3))_P \mid j = 1, …, n/2\}$.

For example, assume that $\ell(x_1) + \ell(x_2) = 5 + 11$, and $\ell(y_1) + \ell(y_2) = -7 + 9$, then there is $\ell(x_1) + \ell(x_2) \neq \ell(y_1) + \ell(y_2)$. Therefore, among $\ell(1)$, …, and $\ell(\tilde{n})$, there does not necessarily exist $\ell(x_1) + \ell(x_2) = \ell(y_1) + \ell(y_2)$.

The above example illustrates that to determinate the existence of $\ell(x_1) + \ell(x_2) = \ell(y_1) + \ell(y_2)$, an adversary must first determinate the constitution of Ω, which will take $O((2^3 3!)^{n/2})$ running time.

5.2.2 Eliminating *W* through the $\|W\|$-th Power

Due to $\lceil \lg M \rceil = 384, 464, 544, \text{or } 640$, $\bar{M}$ can be factorized in tolerable subexponential time. Again due to $\prod_{i=1}^{k} p_i^{e_i} \mid \bar{M}$ and $\prod_{i=1}^{k} e_i \geq 2^{10}$ with $p_k < \tilde{n}/2$, $\|W\|$ can be divined in about 2^{10} running time.

Raising either side of $C_i \equiv (A_i W^{\ell(i)})^\delta$ % M to the $\|W\|$-th power yields

$$C_i^{\|W\|} \equiv (A_i)^{\delta \|W\|} \text{ (\% } M).$$

Let $C_i \equiv g^{u_i}$ (% M), and $A_i \equiv g^{v_i}$ (% M), where g is a generator of $(\mathbb{Z}_M^*, \cdot)$. Then

$$u_i \|W\| \equiv v_i \|W\| \delta \text{ (\% } \bar{M})$$

for $i = 1, …, \tilde{n}$. Notice that $u_i \neq v_i \delta$ (% $\bar{M}$) owing to $\|W\| \mid \bar{M}$.

The above congruence looks to be the MH transform [7]. Actually, $\{v_1\|W\|, \ldots, v_{\bar{n}}\|W\|\}$ is not a super increasing sequence, and moreover there is not necessarily $\lg(u_i\|W\|) = \lg \bar{M}$.

Because $v_i\|W\| \in [1, \bar{M}]$ is stochastic, the inverse δ^{-1} % $\bar{M}$ not need be close to the minimum $\bar{M}/(u_i\|W\|)$, $2\bar{M}/(u_i\|W\|)$, …, or $(u_i\|W\| - 1)\bar{M}/(u_i\|W\|)$. Namely δ^{-1} may lie at any integral position in the interval $[k\bar{M}/(u_i\|W\|), (k+1)\bar{M}/(u_i\|W\|)]$, where $k = 0, 1, \ldots, u_i\|W\| - 1$, which illustrates the accumulation points of minima do not exist. Further observing, in this case, when i traverses the interval $[2, \bar{n}]$, the number of intersections of the intervals including δ^{-1} is likely the max of $(u_2\|W\|, \ldots, u_{\bar{n}}\|W\|)$ which is promisingly close to $\bar{M}$. Therefore, the Shamir attack by the accumulation point of minima is fully ineffectual [16].

Even though find out δ^{-1} by the Shamir attack method, because each of v_i has $\|W\|$ solutions, the number of potential sequences $\{g^{v_1}, \ldots, g^{v_{\bar{n}}}\}$ is up to $\|W\|^{\bar{n}}$. Because of needing to verify whether $\{g^{v_1}, \ldots, g^{v_{\bar{n}}}\}$ is a coprime sequence for each different sequence $\{v_1, \ldots, v_{\bar{n}}\}$, the number of coprime sequences is in direct proportion to $\|W\|^{\bar{n}}$. Hence, the initial $\{A_1, \ldots, A_{\bar{n}}\}$ cannot be determined in subexponential time. Further, the value of W cannot be computed, and the values of $\|W\|$ and δ^{-1} cannot be verified in subexponential time, which indicates that MPP can also be resistant to the attack by the accumulation point of minima.

Additionally, an adversary may divine value of A_i in about ¦Λ¦ running time, where $i \in [1, \bar{n}]$, and compute δ by $u_i\|W\| \equiv v_i\|W\|\delta$ (% $\bar{M}$).

However, because of $\|W\| \mid \bar{M}$, the equation will have $\|W\|$ solutions. Therefore, the running time of finding the original δ is at least

$$\begin{aligned} Ŧ &= \bar{n}¦\Lambda¦L_M[1/3, 1.923] + 2^{10}¦\Lambda¦\|W\| \\ &= \bar{n}¦\Lambda¦L_M[1/3, 1.923] + 2^{10}¦\Lambda¦2^{n-20} \\ &\approx \bar{n}¦\Lambda¦L_M[1/3, 1.923] + 2^n > 2^n. \end{aligned}$$

It is at least exponential in n when $80 \le n \le 128$.

5.3 Attack by a Certain Single C_i

Assume that there is only a solitary $C_i = (A_i W^{\ell(i)})^{\delta}$ % M — $i = 1$ for example, and other C_i's ($i = 2, \ldots, \bar{n}$) are unknown for adversaries.

Through divining $A_1 \in \Lambda$ and $\ell(1) \in \Omega$, the parameters W and $\delta \in (1, \bar{M})$ can be computed. Thus, the number of solution $(A_1, \ell(1), W, \delta)$ will be up to ¦Ω¦¦Λ¦$\bar{M}^2 > 2^n$, which manifests that the original $(A_1, \ell(1), W, \delta)$ cannot be determined in some time being subexponential in n.

Evidently, if $g_1 \equiv A_1 W^{\ell(1)}$ (% M) is a constant, solving $C_1 = g_1^{\delta}$ % M for δ is equivalent to the DLP. Factually, g_1 is not a constant, and at present, seeking the original g_1 and δ will take at least $O(M) > O(2^n)$ steps.

In summary, the time complexity of inferring a related private key from a public key is at least $O(2^n)$.

6 Analysis of Security of a JUNA Plaintext

In this section, we will analyze the security of the new cryptoscheme against recovering a related plaintext from a ciphertext.

The security of a JUNA plaintext depends on the ASPP $\bar{G} \equiv \prod_{i=1}^{n/2} (C_{3(i-1)+B_i})^{B_i}$ (% M) with $C_0 = 1$, but we need also to understand the SPP $\bar{G}_1 \equiv \prod_{i=1}^{n/2} (C_{3(i-1)+B_i})^{\lceil B_i/3 \rceil}$ (% M) with $C_0 = 1$.

Definition 9: Let A and B be two computational problems. A is said to reduce to B in polynomial time, written as $A \le_{\mathrm{T}}^{\mathrm{P}} B$, if there is an algorithm for solving A which calls, as a subroutine, a hypothetical algorithm for solving B, and runs in polynomial time, excluding the time of the algorithm for B [11][17].

The hypothetical algorithm for solving B is called an oracle. It is easy to understand that no matter what the running time of the oracle is, it does not influence the result of the comparison.

$A \le_{\mathrm{T}}^{\mathrm{P}} B$ means that the difficulty of A is not greater than that of B, namely the running time of the fastest algorithm for A is not greater than that of the fastest algorithm for B when all polynomial times are treated as being pairwise equivalent. Concretely speaking, if A cannot be solved in polynomial or subexponential time, B cannot also be solved in corresponding polynomial or subexponential time; and if B can be solved in polynomial or subexponential time, A can also be solved in corresponding polynomial or subexponential time.

Definition 10: Let A and B be two computational problems. If $A \le_{\mathrm{T}}^{\mathrm{P}} B$ and $B \le_{\mathrm{T}}^{\mathrm{P}} A$, then A and B are said to be computationally equivalent, written as $A =^{\mathrm{P}}_{\mathrm{T}} B$ [11][17].

$A =^{\mathrm{P}}_{\mathrm{T}} B$ means that either if A is a hardness of a certain complexity on condition that the dominant variable approaches a large number, B is also a hardness of the same complexity on the identical condition; or A, B both can be solved in linear or polynomial time.

Definition 9 and 10 suggest a reductive proof method called polynomial time Turing reduction (PTR) [17]. Provable security by PTR is substantially relative and asymptotic just as a one-way function is. Relative security implies that the security of a cryptosystem based on a problem is comparative, but not absolute. Asymptotic security implies that even if a cryptosystem based on a problem is proven to be secure, it is practically secure only on condition that the dominant parameter is large enough.

Naturally, we will enquire whether $A <^{\mathrm{P}}_{\mathrm{T}} B$ exists or not. The definition of $A <^{\mathrm{P}}_{\mathrm{T}} B$ may possibly be given theoretically, but the proof of $A <^{\mathrm{P}}_{\mathrm{T}} B$ is not easy in practice.

Let $\hat{H}(y = f(x))$ represent the complexity or hardness of solving the problem $y = f(x)$ for x [15].

6.1 Two Properties

According to Definition 7, the SPP has the following property.

***Property** 7:* The SPP $\bar{G}_1 \equiv \prod_{i=1}^{n/2} (C_{3(i-1)+B_i})^{\lceil B_i / 3 \rceil}$ ($\%$ M) with $C_0 = 1$ is computationally at least equivalent to the DLP in the same prime field, where $B_1 \ldots B_{n/2} \neq 0$ is a bit-pair string.

Proof:

For the explanation of the problem, we extend $B_1 \ldots B_{n/2}$ to a bit string $b'_1 \ldots b'_{\tilde{n}}$ by the following rule for $i = 1, \ldots, n/2$:

① when $B_i = 0$, let $b'_{3(i-1)+1} = b'_{3(i-1)+2} = b'_{3(i-1)+3} = 0$;

② when $B_i \neq 0$, let $b'_{3(i-1)+1} = b'_{3(i-1)+2} = b'_{3(i-1)+3} = 0$ and $b'_{3(i-1)+B_i} = 1$.

Then, we have the equivalent

$$\bar{G}_1 \equiv \prod_{i=1}^{\tilde{n}} C_i^{b'_i} \ (\% \ M).$$

The form of $\bar{G}_1$ here is similar to that of $\bar{G}_1$ in [1].

Besides, define $\bar{G}_1 \equiv \prod_{i=1}^{\tilde{n}} C^{2^{\tilde{n}-i} b'_i} \equiv \prod_{i=1}^{\tilde{n}} (C^{2^{\tilde{n}-i}})^{b'_i}$ ($\%$ M) when $C_1 = \ldots = C_{\tilde{n}} = C$.

Obviously, $\prod_{i=1}^{\tilde{n}} C_i^{b'_i} = L M + \bar{G}_1$. Owing to $L \in [1, \bar{M}]$, deriving the non-modular product $\prod_{i=1}^{\tilde{n}} C_i^{b'_i}$ from $\bar{G}_1$ is infeasible, which means inferring $b'_1 \ldots b'_{\tilde{n}}$ from $\bar{G}_1$ is not a factorization problem.

Assume that $\bar{O}_s(\bar{G}_1, C_1, \ldots, C_{\tilde{n}}, M)$ is an oracle on solving $\bar{G}_1 \equiv \prod_{i=1}^{\tilde{n}} C_i^{b'_i}$ ($\%$ M) for $b'_1 \ldots b'_{\tilde{n}}$.

Let $y \equiv g^x$ ($\%$ M) be of a DLP, where g is a generator of $(\mathbb{Z}^*_M, \cdot)$, and the binary form of x is $\underset{\cdot}{b}_1 \ldots \underset{\cdot}{b}_{\tilde{n}}$, namely $y \equiv \prod_{i=1}^{\tilde{n}} (g^{2^{\tilde{n}-i}})^{\underset{\cdot}{b}_i}$ ($\%$ M).

Then, by calling $\bar{O}_s(y, g^{2^{\tilde{n}-1}}, \ldots, g, M)$, x, namely $\underset{\cdot}{b}_1 \ldots \underset{\cdot}{b}_{\tilde{n}}$ can be found.

According to Definition 9, there is

$$\hat{H}(y \equiv g^x \ (\% \ M)) \leq^{\mathrm{P}}_{\mathrm{T}} \hat{H}(\bar{G}_1 \equiv \prod_{i=1}^{\tilde{n}} C_i^{b'_i} \ (\% \ M)),$$

namely the SPP is at least equivalent to the DLP in the same prime field in complexity. □

See Definition 8, and the ASPP has the following property.

***Property** 8:* The ASPP $\bar{G} \equiv \prod_{i=1}^{n/2} (C_{3(i-1)+B_i})^{\underline{B}_i}$ ($\%$ M) with $C_0 = 1$ is computationally at least equivalent to the DLP in the same prime field, where $\underline{B}_1 \ldots \underline{B}_{n/2}$ is the bit-pair shadow string of $B_1 \ldots B_{n/2} \neq 0$.

Proof:

For the explanation of the problem, we extend $\underline{B}_1 \ldots \underline{B}_{n/2}$ to a bit shadow string $\underline{b}'_1 \ldots \underline{b}'_{\tilde{n}}$ by the following rule for $i = 1, \ldots, n/2$:

① when $\underline{B}_i = 0$, let $\underline{b}'_{3(i-1)+1} = \underline{b}'_{3(i-1)+2} = \underline{b}'_{3(i-1)+3} = 0$;

② when $\underline{B}_i \neq 0$, let $\underline{b}'_{3(i-1)+1} = \underline{b}'_{3(i-1)+2} = \underline{b}'_{3(i-1)+3} = 0$ and $\underline{b}'_{3(i-1)+B_i} = \underline{B}_i$.

Then, we have the equivalent

$$\bar{G} \equiv \prod_{i=1}^{\tilde{n}} C_i^{\underline{b}'_i} \ (\% \ M).$$

The form of $\bar{G}$ here is similar to that of $\bar{G}$ in [1].

Assume that $\bar{O}_a(\bar{G}, C_1, \ldots, C_{\tilde{n}}, M)$ is an oracle on solving $\bar{G} \equiv \prod_{i=1}^{\tilde{n}} C_i^{\underline{b}'_i}$ ($\%$ M) for $\underline{b}'_1 \ldots \underline{b}'_{\tilde{n}}$, where $\underline{b}'_1 \ldots \underline{b}'_{\tilde{n}}$ is the bit shadow string of $b'_1 \ldots b'_{\tilde{n}}$ which corresponds to $B_1 \ldots B_{n/2}$.

Especially, define $\bar{G} \equiv \prod_{i=1}^{\tilde{n}} C^{\tilde{n}^{\tilde{n}-i} \underline{b}'_i} \equiv \prod_{i=1}^{\tilde{n}} (C^{\tilde{n}^{\tilde{n}-i}})^{\underline{b}_i}$ ($\%$ M) with the stipulation $\underline{b}'_i < \tilde{n}$ (namely that $b'_1 \ldots b'_{\tilde{n}}$ contains at least two nonzero bits) when $C_1 = \ldots = C_{\tilde{n}} = C$.

Let $\bar{G}_1 \equiv \prod_{i=1}^{\tilde{n}} C_i^{b'_i}$ ($\%$ M) be of the SPP.

Because $\bar{G}_1 \equiv \prod_{i=1}^{\tilde{n}} C_i^{b'_i}$ ($\%$ M) and $\bar{G} \equiv \prod_{i=1}^{\tilde{n}} C_i^{\underline{b}'_i}$ ($\%$ M) with $0 \leq b'_i \leq \underline{b}'_i$ have the same structure, by calling $\bar{O}_a(\bar{G}_1, C_1, \ldots, C_{\tilde{n}}, M)$, $b'_1 \ldots b'_{\tilde{n}}$ can be found.

According to Definition 9, there is $\hat{H}(\bar{G}_1 \equiv \prod_{i=1}^{\tilde{n}} C_i^{b'_i} (\% M)) \leq_T^P \hat{H}(\bar{G} \equiv \prod_{i=1}^{\tilde{n}} C_i^{\underline{b}'_i} (\% M))$.

Further by transitivity, there is

$$\hat{H}(y \equiv g^x (\% M)) \leq_T^P \hat{H}(\bar{G} \equiv \prod_{i=1}^{\tilde{n}} C_i^{\underline{b}'_i} (\% M)),$$

namely the ASPP is at least equivalent to the DLP in the same prime field in complexity. □

6.2 Resisting LLL Lattice Base Reduction

We know that after a lattice base is reduced through the LLL algorithm, the final reduced base will contain the shortest or approximately shortest vectors, but among them does not necessarily exist the original solution to a subset sum problem because only if

① the solution vector for the SSP is the shortest,

② the shortest vector is unique in the lattice,

will the original solution vector appear in the reduced base with large probability.

In the JUNA cryptoscheme, there are n = 80, 96, 112, or 128 and $\lceil \lg M \rceil$ = 384, 464, 544, or 640. Under this circumstances, the DLP and IFP can be solved in tolerable subexponential time, namely the DLP and IFP can not resist the attack of adversaries.

For convenience, extend $\underline{B}_1 \ldots \underline{B}_{n/2}$ to $\underline{b}'_1 \ldots \underline{b}'_{\tilde{n}}$ by the following rule for $i = 1, \ldots, n/2$:

① when $\underline{B}_i = 0$, let $\underline{b}'_{3(i-1)+1} = \underline{b}'_{3(i-1)+2} = \underline{b}'_{3(i-1)+3} = 0$;

② when $\underline{B}_i \neq 0$, let $\underline{b}'_{3(i-1)+1} = \underline{b}'_{3(i-1)+2} = \underline{b}'_{3(i-1)+3} = 0$ and $\underline{b}'_{3(i-1)+B_i} = \underline{B}_i$.

For example, suppose that $B_1 \ldots B_4$ = 00 10 01 00, then $\underline{B}_1 \ldots \underline{B}_4$ = 0310, and $\underline{b}'_1 \ldots \underline{b}'_{12}$ = 000 030 100 000.

In this way, there is

$$\bar{G} \equiv \prod_{i=1}^{\tilde{n}} C_i^{\underline{b}'_i} (\% M).$$

Let g be a generator of $(\mathbb{Z}_M^*, \cdot)$.

Let $C_1 \equiv g^{u_1} (\% M), \ldots, C_{\tilde{n}} \equiv g^{u_{\tilde{n}}} (\% M)$, and $\bar{G} \equiv g^v (\% M)$.

Then, through a conversion in subexponential time, seeking $\underline{B}_1 \ldots \underline{B}_{n/2}$ from $\bar{G}$ is equivalent to seeking $\underline{b}'_1 \ldots \underline{b}'_{\tilde{n}}$ from the congruence

$$u_1 \underline{b}'_1 + \ldots + u_{\tilde{n}} \underline{b}'_{\tilde{n}} \equiv v (\% \bar{M}), \tag{3}$$

where v may be substituted with $v + k\bar{M}$ along with $k \in [0, \tilde{n} - 1]$ [3].

Similar to Section 1, $\{u_1, \ldots, u_{\tilde{n}}\}$ is called a compact sequence due to every $\underline{b}'_i \in [0, n/4 + 1]$ [4], and solving Equation (3) for $\underline{b}'_1 \ldots \underline{b}'_{\tilde{n}}$ is called an ASSP [1].

This ASSP may also be converted into a SSP, and thus according to $\underline{b}'_i \in [0, n/4 + 1]$, the density of a related ASSP knapsack is defined as

$$\begin{aligned} D &= \sum_{i=1}^{\tilde{n}} \lceil \lg(n/4 + 1) \rceil / \lceil \lg M \rceil \\ &= \tilde{n} \lceil \lg(n/4 + 1) \rceil / \lceil \lg M \rceil. \end{aligned}$$

Namely,

$$D = 3n \lceil \lg(n/4 + 1) \rceil / (2 \lceil \lg M \rceil). \tag{4}$$

which is slightly different from Formula (2).

Concretely speaking, in the JUNA cryptoscheme, there are

$D = 120 \times 5 / 384 \approx 1.5625 > 1$ for $n = 80$ and $\lceil \lg M \rceil = 384$,

$D = 144 \times 5 / 464 \approx 1.5517 > 1$ for $n = 96$ and $\lceil \lg M \rceil = 464$,

$D = 168 \times 5 / 544 \approx 1.5441 > 1$ for $n = 112$ and $\lceil \lg M \rceil = 544$,

$D = 196 \times 6 / 640 \approx 1.8375 > 1$ for $n = 128$ and $\lceil \lg M \rceil = 640$.

Therefore, Equation (3) represents an ASSP of high density, which indicates that many different subsets will have the same sum, and probability that the original solution vector will occur in the final reduced lattice base is nearly zeroth. Meanwhile, our experiment demonstrates that the original solution vector does not occur in the final reduced base.

6.3 Avoiding Adaptive-chosen-ciphertext Attack

Most of public key cryptoschemes may probably be faced with adaptive-chosen-ciphertext attack [18]. The Cramer-Shoup asymmetric encryption scheme is the first efficient one which is extremely malleable, and proven to be secure against the adaptive-chosen-ciphertext attack using standard cryptographic assumptions [19]. In our scheme, to avoiding adaptive-chosen-ciphertext attack, we have two approaches of which each makes the algorithm be able to produce many different ciphertexts to an identical plaintext.

6.3.1 Bringing a Random Bit String into the Encryption

※ The Adjusted Encryption Algorithm

Parallel to Section 3.2, assume that $(\{C_1, …, C_{\tilde{n}}\}, M)$ is a public key, and $B_1…B_{n/2}$ is the bit-pair string of a plaintext block $b_1…b_n \neq 0$.

```
S1: Set $C_0 \leftarrow 1$, $k \leftarrow 0$, $i \leftarrow 1$, $s \leftarrow 0$.
S2: If $B_i = 00$ then
        let $k \leftarrow k + 1$, $\underline{B}_i \leftarrow 0$
    else
        let $\underline{B}_i \leftarrow k + 1$, $k \leftarrow 0$;
        if $s \neq 0$ then $s \leftarrow i$.
S3: Let $i \leftarrow i + 1$.
    If $i \leq n / 2$ then goto S2.
S4: Randomly produce $r_1…r_{n/2} \in \{0, 1\}^{n/2}$.
S5: $r_s \leftarrow 1$.
    If $k \neq 0$ then let $\underline{B}_s \leftarrow \underline{B}_s + k$.
S6: Compute $\bar{G} \leftarrow \prod_{i=1}^{n/2} (C_{r_i(3(i-1)+B_i) + \neg r_i(3(i - \underline{B}_i) \% n / 2 + B_i)})^{\underline{B}_i} \% M$.
```

Clearly, a ciphertext different from one another will be returned by the above algorithm every time when an identical plaintext is inputted repeatedly.

It is easily understood that contrarily a ciphertext can almost uniquely be decrypted in polynomial time in terms of Section 3.3 and 4.2.

※ The Adjusted Decryption Algorithm

Parallel to Section 3.3, we redesign a corresponding decryption algorithm of which the running time is equivalent to that of the algorithm in Section 3.3.

Assume that $(\{A_1, …, A_{\tilde{n}}\}, W, \delta)$ is a related private key, and $\bar{G}$ is a ciphertext.

Notice that due to $\sum_{i=1}^{n/2} \underline{B}_i = n/2$ and $\ell(3(i-1)+B_i)$ odd (excluding $\ell(0) = 0$), $\underline{k} = \sum_{i=1}^{n/2} \underline{B}_i \ell(3(i-1)+B_i)$ must be even.

```
S1: Compute $Z_0 \leftarrow \bar{G}^{\delta^{-1}} \% M$.
    Set $Z_1 \leftarrow Z_0$, $h \leftarrow 0$.
S2: If $2 \mid Z_h$ then do $Z_h \leftarrow Z_h W^{2(-1)^h} \% M$, goto S2.
S3: Set $B_1…B_{n/2} \leftarrow 0$, $j \leftarrow 0$, $k \leftarrow 0$, $l \leftarrow 0$, $i \leftarrow 1$, $G \leftarrow Z_h$.
S4: If ${A_{3i-j}}^{l+1} \mid G$ then let $l \leftarrow l + 1$, goto S4.
S5: Let $j \leftarrow j + 1$. If $l = 0$ and $j \leq 2$ then goto S4.
S6: If $l = 0$ then
        let $k \leftarrow k + 1$, $i \leftarrow i + 1$
    else
        compute $G \leftarrow G / {A_{3i-j}}^{l}$;
        if $k > 0$ or $l \geq i$ then
            let $B_i \leftarrow 3 - j$, $i \leftarrow i + 1$
        else
            let $B_{i+l-1} \leftarrow 3 - j$, $i \leftarrow i + l$;
        set $l \leftarrow 0$, $k \leftarrow 0$.
S7: If $i \leq n / 2$ and $G \neq 1$ then set $j \leftarrow 0$, goto S4.
S8: If $G \neq 1$ then
    set $h \leftarrow \neg h$, do $Z_h \leftarrow Z_h W^{2(-1)^h} \% M$, goto S2.
```

In this wise, the original plaintext block $B_1…B_{n/2}$, namely $b_1…b_n$ is recovered although $r_1…r_{n/2}$ is brought into the encryption process.

It is easy to see that as long as $\bar{G}$ is a true ciphertext, the decryption algorithm can always terminate.

6.3.2 Appending a Stochastic Bit String to a Plaintext

The second approach to avoiding adaptive-chosen-ciphertext attack is to append a stochastic fixed-length bit string to the terminal of a plaintext block when it is encrypted. For a concrete implementation, refer to the OAEP+ scheme [20].

Of course, we may combine the first approach with the second approach.

6.4 Avoiding Meet-in-the-middle Attack

Meet-in-the-middle dichotomy was first developed in 1977 [21]. Section 3.10 of [11] puts forward a meet-in-the-middle attack on a subset sum problem.

```
INPUT: a set of positive integers $\{C_1, C_2, \ldots, C_n\}$ and a positive integer $s$.
OUTPUT: $b_i \in \{0, 1\}$, $1 \le i \le n$, such that $\sum_{i=1}^{n} C_i b_i = s$, provided such $b_i$ exist.
S1: Set $t \leftarrow \lfloor n / 2 \rfloor$.
S2: Construct a table with entries $(\sum_{i=1}^{t} C_i b_i, (b_1, b_2, \ldots, b_t))$ for $(b_1, b_2, \ldots, b_t) \in (\mathbb{Z}_2)^t$.
    Sort this table by the first component.
S3: For each $(b_{t+1}, b_{t+2}, \ldots, b_n) \in (\mathbb{Z}_2)^{n-t}$, do the following:
    S3.1: Compute $r = s - \sum_{i=t+1}^{n} C_i b_i$ and check, using a binary search,
          whether $r$ is the first component of some entry in the table;
    S3.2: If $r = \sum_{i=1}^{t} C_i b_i$, then return (a solution is $(b_1, b_2, \ldots, b_n)$).
S4: Return (no solution exists).
```

It is not difficult to understand that the running time of the above algorithm is $O(n2^{n/2})$.

Likewise, meet-in-the-middle dichotomy may be used to attack the ASSP $\bar{G} \equiv \prod_{i=1}^{n/2} (C_{3(i-1)+B_i})^{\bar{B}_i}$ (% M) when $B_{n/4} \ne 00$ and $B_{n/2} \ne 00$ which occur with the probability 9 / 16 = 0.5625. The attack task will take $O(n2^{n/2}\lg^2 M)$ bit operations.

Hence, to avoid meet-in-the-middle attack as efficiently as possible, parallel to Section 6.3.1, a random bit string $r_1 \ldots r_{n/2}$ should be brought into the encryption algorithm so as to extend the scope of exhaustive search.

In this case, the ASPP is converted into $\bar{G} \equiv \prod_{i=1}^{n/2} (C_{r_i(3(i-1)+B_i) + \neg r_i(3(i - \underline{B}_i) \,\%\, n/2 + B_i)})^{\bar{B}_i}$ (% M), and moreover, it is not difficult to understand that the running time of the attack task rises to $O(n2^{5n/8}\lg^2 M)$ bit operations. Notice that the probability of success is still 9 / 16.

Concretely speaking,

when $n = 80$ with $\lceil \lg M \rceil = 384$, $\mathcal{T} = n2^{5n/8}\lg^2 M = 2^7 2^{50} 2^{18} = 2^{75}$ bit operations,
when $n = 96$ with $\lceil \lg M \rceil = 464$, $\mathcal{T} = n2^{5n/8}\lg^2 M = 2^7 2^{60} 2^{18} = 2^{85}$ bit operations,
when $n = 112$ with $\lceil \lg M \rceil = 544$, $\mathcal{T} = n2^{5n/8}\lg^2 M = 2^7 2^{70} 2^{20} = 2^{97}$ bit operations,
when $n = 128$ with $\lceil \lg M \rceil = 640$, $\mathcal{T} = n2^{5n/8}\lg^2 M = 2^7 2^{80} 2^{20} = 2^{107}$ bit operations.

If the above time complexities do not satisfy users′ requirement, the users need further appending a stochastic bit string to a plaintext.

7 Conclusion

In the paper, the authors propose a new public key cryptoscheme which includes the key generator, encryption algorithm, and decryption algorithm.

The cryptoscheme builds its own security on the MPP $C_i = (A_i W^{\ell(i)})^{\delta}$ % M with $A_i \in \Lambda$ and $\ell(i)$ from Ω and the ASPP $\bar{G} \equiv \prod_{i=1}^{n/2} (C_{3(i-1)+B_i})^{\bar{B}_i}$ (% M) to which no subexponential time solutions are found [22], and there exist only exponential time solutions so far, utilizes a bit-pair string to decrease the bit-length of the modulus M, exploits a bit-pair shadow string to prevent attack by LLL lattice base reduction, and employs the two approaches of introducing a random bit string into the encryption and appending a stochastic bit string to a plaintext to avoid attack by adaptive-chosen-ciphertext measure and meet-in-the-middle dichotomy.

As n = 80, 96, 112, or 128, there exists $\lceil \lg M \rceil$ = 384, 464, 544, or 640, which assures that when a JUNA ciphertext is converted into an ASSP through a discrete logarithm, the density of a related ASSP knapsack is pretty high, and larger than 1.

There exists contradiction between time and security, so does between space and security, and so does between time and space. We attempt to find a balance which is none other than a delicate thing among time, space, and security.

Acknowledgment

The authors would like to thank the Academicians Jiren Cai, Zhongyi Zhou, Changxiang Shen, Zhengyao Wei, Andrew C. Yao, Binxing Fang, Xicheng Lu, and Guangnan Ni for their important guidance, suggestions, and help.

The authors also would like to thank the Professors Dingyi Pei, Dengguo Feng, Jie Wang, Ronald L. Rivest, Moti Yung, Adi Shamir, Dingzhu Du, Mulan Liu, Huanguo Zhang, Yixian Yang, Xuejia Lai, Xiaoyun Wang, Yupu Hu, Kefei Chen, Dongdai Lin, Jiwu Jing, Rongquan Feng, Ping Luo, Jianfeng Ma, Xiao Chen, Zhenfu Cao, Chao Li, Lei Hu, Lusheng Chen, Wenbao Han, Xinchun Yin, Bogang Lin, Qibin Zhai, Dake He, Hong Zhu, Renji Tao, Bingru Yang, Zhiying Wang, and Quanyuan Wu for their important advice, suggestions, and corrections.

References

[1] S. Su and S. Lü, A Public Key Cryptosystem Based on Three New Provable Problems, *Theoretical Computer Science*, vol. 426-427, Apr. 2012, pp. 91-117.

[2] R. L. Rivest, A. Shamir, and L. M. Adleman, A Method for Obtaining Digital Signatures and Public-key Cryptosystems, *Communications of the ACM*, vol. 21(2), 1978, pp. 120-126.

[3] V. Niemi, A New Trapdoor in Knapsacks, *Proc. of Advances in Cryptology: EUROCRYPT '90*, LNCS 473, Springer-Verlag, Berlin, 1991, pp. 405-411.

[4] G. Orton, A Multiple-Iterated Trapdoor for Dense Compact Knapsacks, *Proc. of Advance in Cryptology: EUROCRYPT '94*, Springer-Verlag, 1994, pp. 112-130.

[5] E. F. Brickell, Solving Low Density Knapsacks, *Proc. of Advance in Cryptology: CRYPTO '83*, Plenum Press, 1984, pp. 25-37.

[6] M. J. Coster, A. Joux, B. A. LaMacchia etc, Improved Low-Density Subset Sum Algorithms, *Computational Complexity*, vol. 2(2), 1992, pp. 111-128.

[7] R. C. Merkle and M. E. Hellman, Hiding information and Signatures in Trapdoor Knapsacks, *IEEE Transactions on Information Theory*, vol. 24(5), 1978, pp. 525-530.

[8] S. Y. Yan, *Number Theory for Computing* (2nd ed.), Springer-Verlag, Berlin, 2002, ch. 1.

[9] L. Fibíková and J. Vyskoč, *Practical Cryptography - The Key Size Problem: PGP after Years*, http://www.vaf.sk/download/keysize.pdf, Dec. 2001.

[10] T. W. Hungerford, *Algebra*, Springer-Verlag, New York, 1998, ch. 1-3.

[11] A. J. Menezes, P. van Oorschot, and S. Vanstone, *Handbook of Applied Cryptography*, CRC Press, London, UK, 2001, ch. 2, 3, 8.

[12] D. Naccache and J. Stern, A new public key cryptosystem, *Proc. of Advances in Cryptology: EUROCRYPT '97*, Springer-Verlag, 1997, pp. 27-36.

[13] T. ElGamal, A Public-key Cryptosystem and a Signature Scheme Based on Discrete Logarithms, *IEEE Transactions on Information Theory*, vol. 31(4), 1985, pp. 469-472.

[14] I. F. Blake, G. Seroussi, and N. P. Smart, *Elliptic Curves in Cryptography*, Cambridge University Press, Cambridge, UK, 1999.

[15] M. Davis, *The Undecidable: Basic Papers on Undecidable Propositions, Unsolvable Problems and Computable Functions*, Dover Publications, Mineola, 2004.

[16] A. Shamir, A Polynomial Time Algorithm for Breaking the Basic Merkle-Hellman Cryptosystem, *Proc. of the 23th IEEE Symposium on the Foundations of Computer Science*, IEEE, 1982, pp. 145-152.

[17] D. Z. Du and K. Ko, *Theory of Computational Complexity*, John Wiley & Sons, New York, 2000, ch. 2, 3.

[18] D. Bleichenbacher, Chosen Ciphertext Attacks Against Protocols Based on the RSA Encryption Standard PKCS #1, *Proc. of Advance in Cryptology: Crypto '98*, Springer-Verlag, 1998, pp. 1-12.

[19] R. Cramer and V. Shoup, A Practical Public Key Cryptosystem Provably Secure against Adaptive Chosen Ciphertext Attack, *Proc. of Advance in Cryptology: Crypto '98*, Springer-Verlag, 1998, pp. 13-25.

[20] V. Shoup, OAEP Reconsidered, *Proc. of Advance in Cryptology: Crypto '01*, Springer-Verlag, 2001, pp. 239-259.

[21] W. Diffie and M. E. Hellman, Exhaustive Cryptanalysis of the NBS Data Encryption Standard, *Computer*, vol. 10 (6), 1977, pp. 74-84.

[22] S. Su and S. Lü, REESSE1+ · Reward · Proof by Experiment on 80-bit Moduli, *http://arxiv.org/pdf/0908.0482*, Aug. 2009 (revised Dec. 2012).