An Efficient Code Based Digital Signature Algorithm

Fang Ren[1], Dong Zheng[1], WeiJing Wang[1]
(Corresponding author: Fang Ren)
School of Telecommunication and Information Engineering, Xi'an University of Posts and Telecommunications[1]
Xi'an 710121, China
(Email: renfang_81@163.com)
(Received Aug. 4, 2016; revised and accepted Nov. 15, 2016 & Feb. 19, 2017)

Abstract

In the case of most current digital signature algorithm can be attacked by quantum algorithm, code based digital signature algorithm, which represents the Post-Quantum Cryptography, has become the hotspot of current research. CFS algorithms proposed in 2001 is one of the most important code based digital signature algorithm, but its signature efficiency is very low. In this paper, an improved CFS algorithm is proposed by means of code based hash function. The output of this hash function is a syndrome of a regular word whose weight is no more than error correcting capacity t of the code. By using this hash function instead of the random hash function, the decoding algorithm can avoid the time-consuming syndrome decoding attempts. The signing time of the improved algorithm reduces $t!$ times than the original. At the same time, the signature efficiency is no longer restricted to error correcting capacity of the code. Furthermore, the securities of these two algorithms both rely on the equivalent NP complete problems.

Keywords: Digital Signature; Hash Function; Quantum Attack; Syndrome

1 Introduction

Public-key cryptography has obtained a lot of valuable results since it was developed over 30 years ago. No matter in the field of individual privacy, commercial confidentiality, or even national security etc., it has played a key role. Under the threat of Quantum algorithm [13, 24], most of the widely using public-key algorithms based on number theoretic difficult problems nowadays are no longer secure. Currently, code based public key cryptography technique is regarded as a method which can resist Quantum attack [20, 23]. Because of this, it has become one of the mainstream of public key cryptography in future development.

McEliece proposed the first code-based public-key encryption algorithm based on the irreducible binary Goppa codes [14]. In this algorithm, the encryption process is equivalent to adding a random wrong vector to the plaintext; while the decryption process is corresponding to decoding. Another important algorithm named Niederreiter's algorithm [18] realizes encryption and decryption process through syndrome decoding. It has been proved that its security is equivalent to the McEliece algorithm: that means their security can be reduced to two NP-complete problems: the random binary codes decoding problem and the Goppa code distinguishing problem [3, 9]. Since code based cryptography technology was proposed, there are a number of research achievements over the past 30 years, including encryption, digital signature [6, 22], identification [5], hash function [1], stream ciphers [12] and so on, almost throughout all the fields of cryptography. And in the midst of digital signature field, Courtois-Finiasz-Sendrier (CFS) signature algorithm [6], which was proposed in 2001, has been viewed as a classic algorithm.

CFS algorithm, which is the first secure signature algorithm based on binary Goppa codes, is constructed on the basis of the Niederreiter encryption algorithm. Many comprehensive discussions about the security of the CFS have been made in the past decade [7, 11]. There are also a variety of improved algorithms such as mCFS [7], parallel CFS [10], etc. In addition, other special-purpose signature, such as ring signature [15, 25], blind signatures [19], etc. also can be constructed on the basis of CFS. Similar as CFS, the core of these algorithms is that the hash value of the message has to be transformed to a syndrome of Goppa code through preprocessing during signature. The signing process is decoding syndrome by using the secret decoding algorithm, which regards the codeword as signature value; to verify the validity of signature, the syndrome of the codeword is calculated and compared with the hash value of the message.

Although signature algorithms based on CFS could provide relatively high security, its shortcoming is still evident, namely, the efficiency of signature is rather low.

The original CFS algorithm, for example, in order to get a decodable syndrome, it has to execute $t!$ attempts averagely, where t is error correcting capacity of the Goppa codes. Apparently, if the parameter t increases, signatures times will grow exponentially rapidly, while low security defects can be brought by smaller t value. Meanwhile, this contradicts the basic aim of high error correcting capacity of error correcting codes, which, to some extent, hampers the application of CFS series algorithms.

In this paper, we mainly study the flaw mentioned above of CFS algorithms. Through analyzing algorithm implement details and identifying the main causes of the inefficiency of signature, signing process could be improved. We propose an efficient code based digital signature algorithm, whose signing time does not grow rapidly with the parameter t. Without reducing security, the efficiency of algorithms could be effectively improved. It is a type of more practical digital signature algorithm.

2 Preliminaries

2.1 Error Correcting Codes

Definition 1. *A (n,k) linear code C over a finite field F_q is a linear subspace with dimension k of the vector space F_q^n. The elements of F_q^n are called words, while the elements of C are called codewords. Number n is called the length of C and k is called the rank of it.*

Definition 2. *The matrix $\mathbf{G} \in F_q^{k\times n}$ is a generator matrix for the (n,k) linear code C over F_q, if the row of $\mathbf{G}$ span C over F_q.*

The generator matrix $\boldsymbol{G}$ for linear code C is not unique, but the different generator matrixes can mutually convert by elementary row transformation, namely, if $\boldsymbol{G}$ is a generator matrix for C, and $\boldsymbol{P}$ is an elementary matrix, $\boldsymbol{PG}$ is also a generator matrix for C.

Definition 3. *The parity check matrix $\mathbf{H} \in F_q^{(n-k)\times n}$ of (n,k) linear code C is defined by $\mathbf{H}\cdot x^T = 0, \forall x \in C$.*

The parity check matrix of the linear code is also not unique, which is similar to generator matrix. And different parity check matrix can also mutually convert by elementary row transformation. Vector c of length n is a codeword of C is equivalent to $\boldsymbol{H}c^T = 0$. For any word c, $\boldsymbol{H}c^T$ is called the syndrome of c.

Definition 4. *The Hamming distance $d(u,v)$ is defined as the number of different components of u and v, of which $u = (u_1, u_2, \ldots, u_n)$ and $v = (v_1, v_2, \ldots, v_n)$ are two codewords of the linear codes respectively, i.e. $d(u,v) = |\{i|u_i \neq v_i\}|$. Hamming weight $w(u)$ of codeword u is defined as the Hamming distance between u and all zero codeword, i.e.$w(u) = d(u,0)$, the minimum Hamming weight of non all zero codewords of code C is called the minimum distance of code C, generally sign as d_{min}.*

The error correcting capacity of the code is determined by the minimum distance, in general, error correcting capacity t of linear codes with the minimum distance d_{min} meets the condition $t \leq \lfloor \frac{d_{min}-1}{2} \rfloor$.

Goppa code is a kind of special linear code [2], whose parameters used in the McEliece encryption algorithm have the following form: $n = 2^m, k = n - mt$. The foundation of the efficient decoding is the specific structure of the generator polynomial of Goppa codes, which is also the basis of constructing code based cryptographic algorithms. That is by regarding the structure information of Goppa codes and corresponding decoding algorithm as secret trapdoor information or decryption private keys, a one-way trapdoor function can be used to construct public-key encryption algorithm and signature algorithm.

In this paper, linear codes and Goppa codes are over the binary field F_2.

2.2 Difficult Problems

Public-key cryptography is always founded upon some difficult problems, such as the security of RSA relies on the difficult problem of factoring big integer problem. The following is a summary of some difficult problems on which code based public key algorithms mainly rely. All of them have been proved that are NP-complete problems and can effectively resist known quantum attacks.

Problem 1. *Syndrome Decoding**(SD)** Problem,*

Input: *A finite field F_q, randomly select a matrix $\mathbf{H} \in F_q^{(n-k)\times n}$ and vector $s \in F_q^{n-k}$, integer $k > 0$.*

Output: *A word $x \in F_q^n$, its weight $w(x) \leq k$, and meets $\mathbf{H}x^T = s$.*

Problem 2. *Goppa Codes Distinguishing**(GD)** Problem,*

Input: *A finite field F_q, randomly select a matrix $\mathbf{H} \in F_q^{(n-k)\times n}$.*

Output: *Judge whether $\mathbf{H}$ is a (n,k) Goppa parity check matrix or a (n,k) random code parity check matrix?*

3 CFS Signature Algorithm

3.1 Principle and Realization

Digital signature is an important cryptographic techniques used to realize non-repudiation and authentication. There are generally three different ways to build code based digital signature algorithm: (1) Building an algorithm whose procedure is just the inverse process of the code based public-key encryption algorithm; (2) Using zero-knowledge identification algorithm together with the Fiat-Shamir paradigm to develop a signature algorithm; (3) Constructing a special subset of the syndrome space as the foundation of digital signature algorithm.

CFS signature algorithm belongs to the first category, which is a kind of signature algorithm based on classic

Niederreiter encryption algorithm. Such algorithms digital signature process can be concluded as follows:

- Calculate the hash value of the message m by using a public hash function;
- Regard the hash value as the cipher text and use the signers private key to decrypt it;
- Attach the proper forms of the decryption results behind a message m as a signature value.

For code based signature algorithm, however, it's pretty hard to accomplish the second step. The main reason is the output of cipher text by Niederreiter algorithm should be a syndrome with low weight error vectors. But the message m may not be transferred to a required syndrome, which is the cause of ineffectively decoding. Only the syndrome of the error vector whose weight does not exceed the decoding capacity t of the selected Goppa codes can be decoded successfully. Therefore, in effect, CFS algorithm is a probabilistic signature algorithm, which could not pause transforming the hash value of the message repeatedly until a valid syndrome has been found.

Basic CFS signature algorithm uses an increment counter to tag the number of decoding attempts. In order to avoid the security risks of this counter, Dallot developed a mCFS algorithm [7] which based on CFS signature algorithms but much secure. mCFS includes three phases: ***Gen_mCFS***, ***Sign_mCFS*** and ***Verify_mCFS***. Detailed description of the algorithm is shown in Algorithm 1.

Algorithm 1 mCFS Signature Algorithm

Gen_mCFS

Select a (n,k) Goppa code C randomly over F_2, of which the error correcting capacity is t, and the parity check matrix is $\boldsymbol{H}$, select a valid syndrome decoding algorithm γ;

Select a $(n-k)\times(n-k)$ invertible matrix $\boldsymbol{Q}$ over F_2, and a $n\times n$ permutation matrix $\boldsymbol{P}$ randomly;

Select a public secure hash function $h:\{0,1\}^* \rightarrow F_2^{n-k}$;

Define $< h, t, \boldsymbol{H}^{pub} = \boldsymbol{QHP} >$ as public parameters of the system, and $< \boldsymbol{Q}, \boldsymbol{H}, \boldsymbol{P}, \gamma >$ as the users private key.

Sign_mCFS$(msg, \boldsymbol{Q}, \boldsymbol{P}, \gamma)$

For the signer needs to sign a message msg, the signature process is as follows:

Calculate the hash value of the message msg, $s = h(msg)$;

Randomly select $i \in \{1, 2, \ldots, 2^{n-k}\}$, by using the secret decoding algorithm γ to try to decode $s_i = \boldsymbol{Q}^{-1}h(s||i)$, until i_0 has been found, which meets the existence of $\gamma(s_i)$;

If $v = \gamma(s_{i_0})$, the signature value is $(i_0||v\boldsymbol{P})$.

Verify_mCFS$(msg, i, u, \boldsymbol{H}^{pub})$

Set $< msg, i||u >$ as the message-signature pair of the receiver, the verify process is:

Calculate $a = h(h(msg)||i)$ as well as $b = \boldsymbol{H}^{pub}u^T$;

Signature is valid if and only if $a = b$.

3.2 Performance Analysis

Dallot et has conducted a rigorous formal proof of CFS and mCFS signature algorithm, that the security of the algorithm is reduced to the SD and GD problem under the Random Oracle model. Because of the high level of the security, most of the current code based signature scheme is designed on the basis of CFS.

Even though mCFS algorithm has very high level security, its realization efficiency, namely the speed of signature, is rather low, which is caused by too many syndrome decoding attempts. The analysis of mCFS signature algorithms success probability as below:

For pre-selected Goppa codes $(n = 2^m, k = n - mt)$, we assume that the number of decodable syndrome is N_d, the number of overall syndrome is N_t, obviously

$$N_t = 2^{n-k} = 2^{mt} = n^t \tag{1}$$

The weight of error vector which has decodable syndrome has to be less than the error correcting capacity t, hence

$$N_d = \sum_{i=0}^{t} \binom{n}{i} \approx \binom{n}{t} \approx \frac{n^t}{t!} \tag{2}$$

And the approximate success probability of mCFS signature algorithms is

$$P_s = \frac{N_d}{N_t} \approx \frac{\frac{n^t}{t!}}{n^t} = \frac{1}{t!} \tag{3}$$

That is to say, every $t!$ times attempts can only get one decodable syndrome. With t increasing, this number could grow relatively fast, such as set $t = 10$, a signature can be obtained after trying $10! = 3628800$ times averagely. In some earliest literatures [6] authors proposed $t = 9$. But under Bleichenbacher's attack [11], this parameter is no longer safe, the parameter $m = 15$, $t = 12$ or $m = 16, t = 10$ is recommended. In the long term, with new attack methods proposed, value of t will unavoidably growing larger and larger. In order to obtain a valid signature, the signing speed becomes lower and lower with numbers of syndrome decoding attempts growing exponentially, and at the same time, implement efficiency would become worse.

The main reason of the inefficiency of mCFS signature algorithms is generally the s_i calculated from hash value of the message is not a decodable syndrome of liner code C. In order to decode successfully, it has to find a decodable s_i through trying so many different s_i. A valid decoding and successful signature based on finding a proper s_i which is exactly within decoding capacity of C. In order to improve the efficiency of signature, the

original algorithm needs to be improved, so that the calculated s_i itself or at least in a great probability should be a decodable syndrome required.

4 Efficient Code Based Digital Signature Algorithm

In this section, we first construct a code based hash function and then on the basis of it, we improve the mCFS signature algorithm to obtain an efficient code based digital signature algorithm.

4.1 Code Based Hash Functions

The method for constructing a code based hash function is first proposed by Augot et, which is based on Merkle-Damgard design principle [16, 8], namely, a compression function f permits to loop calculate the given message several rounds for obtaining an iteration value as the hash value of the message. It can be proved that the security of the hash function constructed in accordance with this method has no less security than the compression function [1]. Bernstein and Meziani et improved the implement efficiency of the original method respectively [4, 17]. Such constructing methods can be concluded as:

Set compression function as f, and the input is s bits, the output is r bits ($r < s$). To derive hash value of the given message msg, it needs to do a number of loop iterations by using function f:

- The first round: Select the initial vector IV of length r; Select $s - r$ bits from a given message msg, sign as m_0, and concatenate it with the IV as the initial input vector of f with length s, then get r bits initial output;
- Starting from the second round, feed r bits pre-round output back to the input, similar as the first round, select $s - r$ bits from the message msg as m_i in order, concatenate r with m_i as the input vector of f. Calculate the new r bits output.
- Loop this process until the message is taken out. During the final round, if the remaining bits of the message msg are insufficient to $s - r$ bits, randomly select some bits to meet the requirement. The final output of the function f is the hash value of the message msg.

Figure 1 shows this iterative process. In the construction of the hash function mentioned above, the compression function f is the most important part, and even the security of hash function also depends on the security of f. A kind of constructing method of compression function f based on coding difficult problem is presented below.

First select a (n, k) Goppa codes, where $n = 2^m, k = n - mt$, and select a positive integer $w|n$. It is clear that $w = 2^{m'}, m' < m$. Set $l = n/w = 2^{m-m'}$.

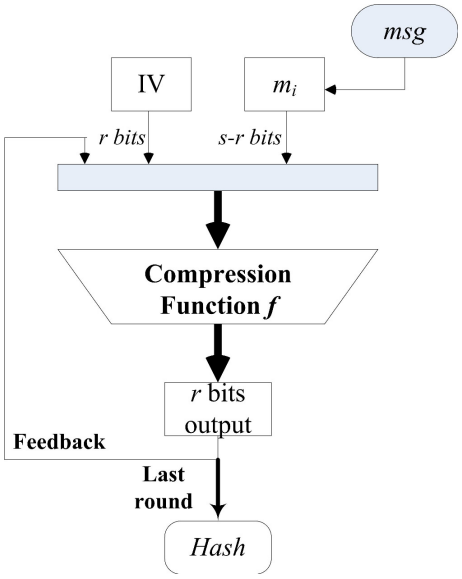

Figure 1: The diagram of hash iterations

For any word c of length n, it can be divided into w blocks of equal length, each block contains l bits. If a word c of weight w within each block $((i-1)l, il]$ happens to have only one 1, c is called as *regular word*.

Set $\boldsymbol{H}$ as the parity check matrix of Goppa codes, which is a $(n - k) \times n$ matrix. Divide $\boldsymbol{H}$ into w submatrix $\boldsymbol{H}_i, i = 1, 2, \ldots, w$ in accordance with the following method

$$\boldsymbol{H} = (\boldsymbol{H}_1, \boldsymbol{H}_2, \ldots, \boldsymbol{H}_w) \tag{4}$$

of which$\boldsymbol{H}_i = (h_{(i-1)l+1}, h_{(i-1)l+2}, \ldots, h_{il})$, and$h_j$ is the jth column of the matrix $\boldsymbol{H}$.

Next we define compression function $f : F_2^s \to F_2^r$, where $s = wlog_2 l$, and $r = n - k = mt$ is the number of matrix $\boldsymbol{H}$'s rows.

For any $x \in F_2^s$, x is divided into the w blocks of equal length in accordance with the same way, that is $x = (x_1, x_2, \ldots, x_w)$, and $x_i \in F_2^{log_2 l}$. Convert x_i into numbers between 0 to $l - 1$. Select the $(x_i + 1)$th column of the matrix $\boldsymbol{H}_i$, that is $h_{(i-1)l+x_i+1}$. Calculate $z = \sum_{i=1}^{w} h_{(i-1)l+x_i+1}$, then the output of the compression function is $f(x) = z$.

Theorem 1. *The output of the compression function f above is equivalent to calculating a syndrome of a regular word of length n and weight w, that is, for any $x \in F_2^s$, a regular word c could be found which meets $\mathbf{H}c^T = f(x)$.*

Proof. First of all, according to the definitions above,

$$f(x) = \sum_{i=1}^{w} h_{(i-1)l+x_i+1}. \tag{5}$$

Define a word $c = (c_1, c_2, \ldots, c_n)$ of length n is as follows:$c_j = 1 \Leftrightarrow \exists x_i, (i-1)l + x_i + 1 = j$. That means existing a x_i, after converting it to a decimal number, the

selected column number is corresponding to the location label j of c_j. Due to calculating a syndrome of a word is equivalent to adding matrix $\boldsymbol{H}$'s columns which are corresponding to non-zero bits of the word, by definition, $f(x)$ is exactly the syndrome of word c, namely $\boldsymbol{H}c^T = f(x)$.

According to the definition of c, c have and only have one 1 within each block $((i-1)l, il], i = 1, 2, \ldots, w$. So that c is a regular word of weight w. □

Based on the compression function above, we define a code based hash function $h_c : \{0,1\}^* \to F_2^r$ as below:

For a given message msg, choose (n, k) Goppa codes and get a compression function f in accordance with the definition above. Through using Augot loop iteration method several times to compress message msg by f, we can obtain a bit string of length r as hash values $h_c(msg)$. The function h_c can apply on arbitrary length message msg, and the output is a bit string of length $r = n - k$.

Theorem 2. *As the above definition, the output of code based hash function h_c is a syndrome of a regular word of length n and weight w.*

Proof. According to the loop iteration constructing methods of the hash function, the output hash value of the final round is also the output of the function f. According to Theorem 1, for any message msg, $h_c(msg)$ is a syndrome of a regular word of length n and weight w. □

We analyze the security of hash functions having this structure: obviously the one-way character of hash function h_c relies on a special SD problem:

Input: A $(n-k) \times n$ matrix $\boldsymbol{H}$ over finite field F_2, vector $s \in F_2^{n-k}$, integer $k > 0$;

Output: A regular word $x \in F_2^n$, its weight $w(x) \leq k$, and satisfies the condition $\boldsymbol{H}x^T = s$.

Augot called this problem as Regular Syndrome Decoding (**RSD**) Problem. It can be proved that this is a NP complete problem [1].

4.2 An Efficient Digital Code Based Signature Algorithm

The mCFS algorithm,which is proposed by Dallot et, can improve the original CFS signature algorithm [7] with stronger security. In this section, by applying the code based hash function h_c given in 4.1, we aim to improve the implement efficiency of mCFS to obtain an efficient signature algorithm mCFS$_c$. This signature algorithm can greatly improve the signature efficiency of mCFS without any decrease of security. mCFS$_c$ also includes three phases: ***Gen_mCFS***$_c$, ***Sign_mCFS***$_c$ and ***Verify_mCFS***$_c$. Algorithm 2 gives the details of the algorithm.

The correctness of verify process in Algorithm 2 can be proved as below: If $< msg, R'||u >$ is a legitimate pair of message-signature through signature process above, it can get the equation as follows:

$$\begin{aligned} b &= \boldsymbol{H}^{pub}u^T = \boldsymbol{HP}(v\boldsymbol{P})^T = \boldsymbol{HPP}^T v^T \\ &= \boldsymbol{H}v^T = s = h_c(h_c(msg)||R') = a \end{aligned}$$

Algorithm 2 mCFS$_c$ Signature Algorithm

1: ***Gen_mCFS***$_c$
2: Selects a (n, k) Goppa codes C randomly over F_2, with the correcting error capacity t and the parity check matrix $\boldsymbol{H}$, a valid decoding syndrome algorithm γ;
3: Randomly select a $n \times n$ permutation matrix $\boldsymbol{P}$ over F_2;
4: Choose a positive integer $w \leq t$ and $w|n$, and construct code based hash function $h_c : \{0,1\}^* \to F_2^{n-k}$;
5: Define $< h_c, t, \boldsymbol{H}^{pub} = \boldsymbol{HP} >$ as system public parameters, and $< \boldsymbol{H}, \boldsymbol{P}, \gamma >$ as the users private key.
6: ***Sign_mCFS***$_c(msg, \boldsymbol{P}, \gamma)$
7: Set the message of the signer is msg, and the signature process is:
8: Choose a one-time random number $R \in \{1, 2, \ldots, 2^{n-k}\}$, and calculate $s = h_c(h_c(msg)||R)$;
9: Set $v = \gamma(s)$, so that the signature value is $(R||v\boldsymbol{P})$.
10: ***Verify_mCFS***$_c(msg, R', u, \boldsymbol{H}^{pub})$
11: Set the received message signature pair is $< msg, R'||u >$, the verify process is:
12: Calculate $a = h_c(h_c(msg)||R')$ and $b = \boldsymbol{H}^{pub}u^T$;
13: Signature is valid if and only if $a = b$.

5 Performance Analyses of Algorithms

This section focuses on the security analysis and efficiency analysis of the code based signature algorithm mCFS$_c$ mentioned in Section 4.2 and comparing it with other existing code based signature algorithms.

Between the three kinds of construction method of building code based digital signature algorithm, the second one, based on zero-knowledge identification algorithm and the Fiat-Shamir paradigm, always have very long signature length [20], roughly 120 Kbits. The third method, constructing a special subset of the syndrome space as the foundation of digital signature algorithm, have been proved only be used as one-time signature [20]. So, the first method, represented by mCFS, is the mainstream of code based signature and we only compare our algorithm with the mCFS algorithm.

5.1 Security Analysis

First of all, we analyze the security. Compared with the mCFS signature algorithm, the primary difference is replacing the random hash function h with the code based hush function h_c. The point is the essence of this change is that it substitutes random hash function for a trapdoor hash function, and the trapdoor information is decoding

Table 1: The security comparison of two algorithms

Signature algorithm	Dependent problems	Hardness of problems
mCFS	SD, GD	NP complete
mCFS_c	RSD, GD	NP complete

Table 2: The efficiency comparison of two algorithms

Signature algorithm	Hash times	Decoding times	Hash times($t=9$)	Decoding times($t=9$)
mCFS	$t!+1$	$t!$	362881	362880
mCFS_c	2	1	2	1

Table 3: The signature time consumption (in seconds) of two algorithms

(m,t)	(15,7)	(15,8)	(15,9)	(16,7)	(16,8)	(16,9)
mCFS	189.58	2570.48	35562.24	442.51	7862.41	57697.92
mCFS_c	0.052	0.073	0.109	0.096	0.203	0.327

algorithm γ of selected Goppa codes. For this type of hash functions, anyone who knows the trapdoor information can effectively calculate the inverse of the hash value, or else, any useful values cannot be provided without the trapdoor information.

In mCFS_c, decoding algorithm γ is the signer's private key which couldn't be obtained but the signer. The security of this hash function can be guaranteed so long as the absolute confidentiality of private key. So the security of mCFS_c is equivalent to mCFS. Hence, this change does not result in any reduction of security. Table 1 shows the security comparison of these two algorithms.

5.2 Efficiency Analysis

According to Algorithm 2, during the process of signing message msg, it has to perform twice hash computation and once syndrome decoding algorithm. According to the Theorem 2, the output of hash functions h_c is a syndrome of a regular word of weight w which does not exceed decoding capacity t of the selected Goppa codes. Therefore anyone who has the secret syndrome decoding algorithm γ can always effectively obtain one regular word of length n and weight w. Compared with mCFS algorithm average $t!$ times attempts to get a decodable syndrome, the biggest advantage of mCFSc is greatly improving signature speed by relieving plenty of decoding attempts. In the long term, this algorithm provides a fundamental method to liberate algorithm from the restriction of code parameter t, so that we can obtain high security by choosing very large t without any reduction of signature speed. Table 2 shows the efficiency comparison of these two algorithms.

In Table 2, parameter t takes the classical value 9. In order to obtain higher security, this value should be increased, and $t = 10$ or $t = 12$ is recommended [11]. It is easy to see with t increases, the consumption of mCFS will increase rapidly, while the consumption of our algorithms mCFS_c remains very low. In order to resist the new attacks in the future, the value of t will unavoidably growing larger and larger, and the implement efficiency of mCFS will become worse and worse, while mCFS_c always has good performance.

5.3 Experimental Results

In this section, we give some experimental results to reveal the efficiency difference between mCFS and mCFS_c. Because of the similarity of ***Gen*** and ***Verify*** phases of these two algorithms, we only count the time consumption of ***Sign*** phase, the most time-consuming phase in Algorithm 1 and Algorithm 2.

The software we used is Magma V2.12, running on 64 bit Windows7 operating system, and the hardware parameters are: Intel Core i7-4710, 2.50GHz, 4GB RAM. The decoding algorithm for Goppa codes is the *Patterson* algorithm [21].

We first selected six different Goppa codes with different parameters m and t. For each code we selected 20 text files with size of 10kB and counted the average time consumption of ***Sign*** phase in these two different algorithms. The experimental results are show in Table 3.

6 Conclusions

As the most important code based digital signature algorithm, the security and implement efficiency of CFS has been extensively studied since it was first proposed. However, with parameter increasing very quickly, its still hard to fundamentally solve the sharp reduction of the signing speed. The further application of the algorithm is therefore seriously limited.

This article proposed and analyzed an improved code based signature algorithm $mCFS_c$ by introducing code based hash function into mCFS algorithm. $mCFS_c$ algorithm can be expected to avoid repeated decoding syndrome attempts to find a decodable syndrome, which increases the signature speed. In addition, compared with mCFS, the signature time can be greatly reduced without any reduction of error correcting capacity t. Meanwhile, the new method has the same security as mCFS algorithm. Therefore it is a more practical code based signature algorithm.

Acknowledgments

This paper was supported by the National Nature Science Foundation of China (Program No. 61272037, 61472472, 41504115); Natural Science Basic Research Plan in Shaanxi Province of China (Program No. 2015JQ6262, 2016JM6033) and Scientific Research Program Funded by Shaanxi Provincial Education Department (Program No.15JK1669, 15JK1661). The authors gratefully acknowledge the anonymous reviewers for their valuable comments.

References

[1] D. Augot, M. Finiasz, and N. Sendrier., "A family of fast syndrome based cryptographic hash functions," in *Progress in Cryptology (Crypto'05)*, pp. 64–83, 2005.

[2] E. Berlekamp, "Goppa codes," *IEEE Transactions on Information Theory*, vol. 19, no. 5, pp. 590–592, 1973.

[3] E. R. Berlekamp, R. J. McEliece, and H. C. A. Van Tilborg, "On the inherent intractability of certain coding problems," *IEEE Transactions on Information Theory*, vol. 24, no. 3, pp. 384–386, 1978.

[4] D. J. Bernstein, T. Lange, and et al. C. Peters, "Really fast syndrome-based hashing," in *Progress in Cryptology (AFRICACRYPT'11)*, pp. 134–152, 2011.

[5] P. L. Cayrel and P. Véron, "Improved code-based identification scheme," *Computer Science*, arXiv:1001.3017, 2010. (`https://arxiv.org/abs/1001.3017`)

[6] N. T. Courtois, M. Finiasz, and N. Sendrier, "How to achieve a mceliece-based digital signature scheme," in *Advances in Cryptology (ASIACRYPT'01)*, pp. 157–174, 2001.

[7] L. Dallot, "Towards a concrete security proof of courtois, finiasz and sendrier signature scheme," in *Research in Cryptology*, pp. 65–77, Berlin Heidelberg: Springer, 2008.

[8] I. B. Damgard, "A design principle for hash functions," in *Advances in Cryptology (CRYPTO'89)*, pp. 416–427, Springer New York, 1990.

[9] D. Engelbert, R. Overbeck, and A. Schmidt, "A summary of mceliece-type cryptosystems and their security," *Journal of Mathematical Cryptology*, vol. 1, no. 2, pp. 1–51, 2007.

[10] M. Finiasz, "Parallel-cfs," in *Selected areas in cryptography*, pp. 159–170, 2011.

[11] M. Finiasz and N. Sendrier, "Security bounds for the design of code-based cryptosystems," in *Advances in Cryptology (ASIACRYPT'09)*, pp. 88–105, Berlin Heidelberg: Springer, 2009.

[12] P. Gaborit, C. Lauradoux, and N. Sendrier, "Synd: a fast code-based stream cipher with a security reduction," in *IEEE International Symposium on Information Theory*, pp. 186–190, Nice, France: IEEE Information Theory Society, 2007.

[13] L. K. Grover, "A fast quantum mechanical algorithm for database search," *Annual Acm Symposium on Theory of Computing*, pp. 212–219, 1996.

[14] R. J. McEliece, "A public-key cryptosystem based on algebraic coding theory," *DSN progress report*, vol. 42, no. 44, pp. 114–116, 1978.

[15] C. A. Melchor, P. Cayrel, and et al. P. Gaborit, "A new efficient threshold ring signature scheme based on coding theory," *IEEE Transactions on Information Theory*, vol. 57, no. 7, pp. 4833–4842, 2011.

[16] R. C. Merkle, "One way hash functions and des," in *Advances in Cryptology (CRYPTO'89)*, pp. 428–446, Springer New York, 1990.

[17] M.Meziani, O.Dagdelen, and et al. P. L.Cayrel, "S-fsb: An improved variant of the fsb hash family," *International Journal of Advanced Science and Technology*, vol. 35, pp. 73–82, 2011.

[18] H. Niederreiter, "Knapsack-type cryptosystems and algebraic coding theory," *Problems of Control and Information Theory*, vol. 15, no. 2, pp. 159–166, 1986.

[19] R. Overbeck, "A step towards qc blind signatures," *Iacr Cryptology Eprint Archive*, 2009.

[20] R. Overbeck and N. Sendrier, "Code-based cryptography," in *Post-quantum cryptography*, pp. 95–145, 2009.

[21] N. Patterson, "The algebraic decoding of goppa codes," *IEEE Transactions on Information Theory*, vol. 21, no. 2, pp. 203–207, 1975.

[22] M. K. Preetha, V. Sachin, and R. C. Pandu, "On provably secure code-based signature and signcryption scheme," *Iacr Cryptology Eprint Archive*, 2012.

[23] F. Ren, D. Zheng, and J. Fan, "Survey of digital signature technology based on error correcting codes(in chinese)," *Chinese Journal of Network and Information Security*, vol. 2, no. 11, pp. 1–10, 2016.

[24] P. W. Shor, "Algorithms for quantum computation: discrete logarithms and factoring," *Symposium on Foundations of Computer Science*, pp. 124–134, 1994.

[25] D. Zheng, X. Li, and K. Chen, "Code-based ring signature scheme," *International Journal of Network Security*, vol. 5, no. 2, pp. 154–157, 2007.

Biography

Fang Ren received his PhD degree in cryptography from Xidian University in 2012. Now he is an associate professor of Xi'an University of Posts and Telecommunications. His research interests include information security and code based cryptography.

Dong Zheng received his PhD degree from Xidian University in 1999. Now he is a professor of National Engineering Laboratory for Wireless Security,Xi'an University of Posts and Telecommunications. His research interests include cloud security, code based systems and other new cryptographic technology.

Weijing Wang received her Master degree in information security from Xi'an University of Posts and Telecommunications in 2017. Her research interests include information security and biometrics protection.