Open Mathematics

Research Article

Kirill Morozov, Partha Sarathi Roy, Rainer Steinwandt, and Rui Xu*

On the security of the Courtois-Finiasz-Sendrier signature

https://doi.org/10.1515/math-2018-0011
Received April 13, 2017; accepted December 19, 2017.

Abstract: We prove that a variant of the Courtois-Finiasz-Sendrier signature is strongly existentially unforgeable under chosen message attack in the random oracle model, assuming hardness of the Permuted Goppa Syndrome Decoding Problem (also known as the Niederreiter problem). In addition, we explicitly show that security against key substitution attacks can be arranged by a standard technique of Menezes and Smart, hashing the public key.

Keywords: Post-quantum cryptography, Code-based digital signature, Courtois-Finiasz-Sendrier signature, Provable security

MSC: 94A60, 68P30

1 Introduction

Post-quantum cryptography is concerned with classical cryptographic solutions that offer security in the presence of large-scale quantum computers. The importance of this line of research beyond the academic setting, is evidenced by NIST's interest in working toward a standardization in this area [1]. Digital signatures [2] are one area of particular interest, and several proposals building on coding theory have been made—for a survey we refer to Cayrel and Meziani [3] (see also [4]).

Currently, there are two major directions in the literature, both invoking the random oracle model: the FDH-like scheme by Courtois, Finiasz and Sendrier [5] (we will refer to it as the CFS signature) and the signatures based on identification schemes using the Fiat-Shamir (FS) transform (see, e.g., [6]). The signature size of the former scheme is much smaller compared to that of the latter. The CFS scheme was proven existentially unforgeable under chosen message attack (EUF-CMA) by Dallot [7] under hardness of the Goppa-Parametrized Bounded Decoding (GPBD) problem and the Goppa-Code Distinguishing (GD) problem. Regrettably, Faugère et al. [8] showed the GD problem to be solvable in polynomial time for the parameters related to the CFS signature by presenting, in particular, a distinguisher for Goppa codes of high rate. This disproves the hardness of the GD problem for the parameters relevant to the CFS scheme, and hence invalidates Dallot's security argument.

Kirill Morozov: Department of Computer Science and Engineering, University of North Texas, USA,
E-mail: Kirill.Morozov@unt.edu (The work was done in part when the author was with Tokyo Institute of Technology, Japan)
Partha Sarathi Roy: Information Security Group, KDDI Research, Inc., Japan, E-mail: royparthasarathi0@gmail.com, pa-roy@kddi-research.jp
Rainer Steinwandt: Department of Mathematical Sciences, Florida Atlantic University, USA, E-mail: rsteinwa@fau.edu
***Corresponding Author: Rui Xu:** School of Computer Science, China University of Geosciences, China,
E-mail: xurui@cug.edu.cn

1.1 Our contribution

Revised security analysis

In this paper, we adjust the security argument for the CFS scheme by observing that the distinguisher does not solve the underlying decoding problem, yet this problem is exactly the one that needs to be solved to obtain an existential forgery. Since the assumption of GD to be hard is no longer true, we use a stronger assumption—the assumed hardness of the Permuted Goppa Syndrome Decoding (PGSD) Problem (see Section 2 for the definition). It is sometimes referred to as the Niederreiter Problem. In fact, we prove that the CFS scheme is *strongly* existentially unforgeable under chosen message attacks (SEUF-CMA). We justify strengthening the assumption by observing that the problem, which the adversary is facing in creating an existential forgery, is indeed a decoding problem, while indistinguishability of the public key is not a concern in the context of (standard) digital signatures. This problem is related to one-wayness of the code-based Niederreiter public-key encryption [9], and it is a long-standing open problem [10].

We note that Dallot's original proof claiming EUF-CMA security can be fixed by directly plugging the new assumption into it. In comparison, we show that the CFS signature is *strongly* EUF-CMA secure. Moreover, our proof allows us to make a precise security statement. Finally, using a technique by Coron [11], we obtain a tighter reduction in the proof compared to that of Dallot. However, our reduction is not as tight as Coron's result for an RSA-based signature: The RSA function can be viewed as a random permutation such that each element in the range has a pre-image in the domain. In comparison, when it comes to the syndrome decoding problem, we cannot guarantee that each element in the range is a valid syndrome (i.e., that it corresponds to an error vector of Hamming weight at most t). This distinction makes the re-randomization trick in Coron's proof inapplicable in our setting.

Security against key substitution attacks

Key substitution attacks target the signature, for which the adversary seeks to compute a new public key for which the signature would verify. Dou et al. [12] presented key substitution attacks against the CFS signature and mentioned the following countermeasure [13] to counter this attack: Pad the message with the public key before hashing. However, this approach does not work in general as pointed out in [14], and the formal proof is not provided in [12]. We give a proof that for the CFS signature this simple countermeasure indeed works.

1.2 Related work

Mathew et al. [15] proposed to mask the public key in the CFS scheme, and provided a heuristic argument that it defeats Faugère et al.'s distinguisher [8], hereby proposing an alternative solution to the one presented in this work. They assume hardness of PGSD and a stronger version of Goppa distinguishability (based on a special coding problem). The disadvantages, compared to our proposal, are the absence of strong EUF-CMA security, an additional (not well-studied) coding assumption, and an additional computational cost incurred by the masking.

2 Preliminaries

Throughout, we denote by $\mathrm{wt}(x)$ the Hamming weight of a vector x. To start out we recall some basic terminology.

2.1 Security of digital signatures

A *digital signature scheme* $\Sigma = (\mathbf{Gen}, \mathbf{Sign}, \mathbf{Verify})$ is comprised of three algorithms.

Gen: Takes a security parameter 1^λ and outputs a pair of keys (sk, pk) which are the signing key and the verification key, respectively. The verification key is public.

Sign: On input a message m and the signing key sk, outputs a signature σ on the message m.

Verify: Takes as input the verification key, a message m and a signature σ, and outputs a bit denoting accept or reject, respectively.

The standard security notion for a signature scheme is *Existential Unforgeability against Chosen Message Attack* (EUF-CMA): The forger gets a public key pk from a challenger who generates a key pair (sk, pk). The forger can query a signing oracle on polynomially many messages m_i hereby obtaining signatures σ_i. The forger can also issue a hash query on a message m and obtains its hash value. We say that the forger wins the EUF-CMA game, if she successfully outputs a pair (m^*, σ^*), where σ^* is a valid signature of a message m^* under the signing key sk with the restriction that the forger has never requested a signature for the message m^*.

For some applications, an upgraded version of the above property, called *Strong Existential Unforgeability against Chosen Message Attack* (SEUF-CMA) is required.[1] The original EUF-CMA definition requires that the adversary cannot forge a signature of a message, which has not been signed by the signer. The SEUF-CMA definition requires that the adversary cannot even produce a different signature on a message which has already been signed. The SEUF-CMA game is almost identical to the EUF-CMA game except that when the forger outputs a valid message-signature pair (m^*, σ^*), she succeeds, if σ^* has never been returned by the challenger as the signature of m^*.

We say that a signature scheme is $(t, \epsilon, q_{hash}, q_{sig})$-SEUF-CMA if the probability of a forger running in time t, who is allowed to query the hash oracle and the signature oracle at most q_{hash} and q_{sig} times, respectively, to win the above game is at most ϵ. The probability distribution is induced by the randomness of both the signature scheme and the internal random coins of the forger.

2.2 Security assumptions

Next, we introduce a problem, which is closely related to the syndrome decoding problem. The difference is that the underlying code is a permuted version of the Goppa code chosen exactly as the public key of the Niederreiter PKE [9].

Permuted Goppa Syndrome Decoding Problem (PGSDP):

Instance: An $(n-k)\times n$ parity check matrix H for a binary Goppa code G capable of correcting up to t errors, a random $(n-k)\times(n-k)$ binary non-singular matrix S and a random $n \times n$ permutation matrix P. Let $H^{pub} = SHP$ and $s = (s_1, \dots, s_{n-k})$.

Problem: Find a binary vector $x = (x_1, \dots, x_n)$ of Hamming weight at most t, such that $H^{pub} \cdot x^T = s$ (we will call the vector s a *syndrome*).

We say that PGSDP is (t, ϵ)-hard, if the probability of finding a solution to a randomly given syndrome is at most ϵ within time t.

2.3 Modified CFS

Let us consider a variant of the CFS signature scheme [5], introduced by Dallot [7], where a randomly chosen value will be concatenated with the message rather than an accumulative counter used in the original CFS signature scheme. We refer to this random variable as a *salt* in the proof of the next section. A difference from Dallot's scheme is that now the public key is also added to the hash (being concatenated with the message

1 This is for instance of interest, when a signature is part of a session identifier in a group key establishment protocol.

and the counter). The latter variant, which we refer to as *mCFS*, is proven to be secure against key substitution attacks as described in Section 4.

Gen$_{mCFS}(1^\lambda)$**:** Select n, k and t according to the security parameter λ. Pick a random parity check matrix H_0 of an (n, k)-binary Goppa code C_0 decoding t errors. This code remains secret. The public key is obtained by randomly permuting the coordinates of C_0 and multiplying by a random non-singular matrix of dimension $n - k$. Specifically, choose a random $(n - k) \times (n - k)$ non-singular matrix U, a random $n \times n$ permutation matrix P and a hash function $h : \{0, 1\}^* \to \mathbb{F}_2^{n-k}$. The public key pk is $H = UH_0P$ and the private key sk is (U, H_0, P). Set $t = \frac{n-k}{\log_2 n}$.

Sign$_{mCFS}(M, H_0)$**:**

1. Choose $r_i \in_R \{0, 1\}^{n-k}$.
2. Compute $x = \text{Decode}_{H_0}(U^{-1}h(M\|pk\|r_i))$.
3. If no x was found, then go to Step 1.
4. Output (r_i, xP).

Verify$_{mCFS}(M, r_i', x', H)$**:** If $\text{wt}(x') > t$, then the signature is invalid. Otherwise, compute $s' = Hx'^T$ and $s = h(M\|pk\|r_i')$. The signature is valid if s and s' are equal.

3 Security against Strong Existential Forgeries

In this section, we prove the SEUF-CMA security of the mCFS signature scheme presented above, based on the assumption that PGSDP is hard. One crucial property that is worth mentioning towards our result is that the errors, which are output by **Sign**$_{mCFS}$ algorithm in the random oracle model, follow the uniform distribution over words of Hamming weight at most t. Note that in this proof, for the sake of simplicity, we have ignored, without loss of generality, the public key inside the hash function.

Theorem 3.1. *Suppose that h is a random oracle, PGSDP is (t', ϵ')-secure, and let $T_s(n, k)$ denote the time to compute a syndrome for an (n, k) Goppa code. Then, the mCFS scheme is $(t, \epsilon, q_{hash}, q_{sig})$-secure*[2]*, where:*

$$t = t' - q_{sig}(q_{hash} - q_{sig})\Theta(n - k) - q_{hash}\Theta(n - k) - q_{sig}T_s(n, k),$$
$$\epsilon \le \frac{2^{n-k}}{2^{n-k} - q_{sig}} \frac{1}{1 - 2^{k-n}} q_{hash}\epsilon'.$$

Proof. Let $\mathcal{F}$ be a forger which $(t, \epsilon, q_{hash}, q_{sig})$-breaks the SEUF-CMA security of the mCFS signature scheme. We construct a decoder $\mathcal{D}$ that (t', ϵ')-solves the PGSDP. The decoder is given input (H, s^*) and tries to find a vector $e \in \mathbb{F}_2^n$ such that $He^T = s^*$ and $\text{wt}(e) \le t$. The decoder will challenge the forger $\mathcal{F}$ and simulate the oracles for it. If the forger succeeds with a valid forgery of the mCFS signature then with non-negligible probability the decoder can use the forgery made by the forger to resolve its given instance of the syndrome decoding problem.

Without loss of generality, we assume that the forger never repeats a hash query. However, the forger may repeat a signature query in order to get different signatures of the same message. The decoder maintains a counter i, initially set to zero.

When a message M appears for the first time in the hash query or the signature query, the decoder increments the counter i and sets $M_i \leftarrow M$. Then the decoder $\mathcal{D}$ generates a list R_i of q_{sig} random values (with repetition) in $\{0, 1\}^{n-k}$ as the pre-defined salts that, when concatenated with M_i, hash to decodable syndromes. We use the expression *decodable salts* to denote all salts that correspond to decodable syndromes. Denote by N_{total} the number of all available salts, and by N_{dec} the number of decodable salts. Clearly $N_{total} = 2^{n-k}$. By the analysis from CFS [5], $N_{dec} \approx N_{total}/t!$. Since we model the hash function as a random

[2] In the NIST specifications, it is mentioned that the number q_{sig} of signing queries can be as large as 2^{64} for a security level of 128 bits. In this case, the factor $q_{sig} \times q_{hash}$ is non-negligible, which shows that one much take extra care about these parameters, when dealing with practical implementations.

oracle, the probability that a salt is decodable is the same for all salts. Thus we can generate the list R_i of length q_{sig} by sampling from all 2^{n-k} salts with each of them being chosen with probability $q_{sig}/(t! \cdot N_{dec})$. The decoder also randomly chooses an integer c from $[1, \dots, q_{hash}]$ so that it answers the c-th hash query with s^*.

When the forger $\mathcal{F}$ makes a hash query for $M_i \| r_i$, if it is the c-th hash query, the decoder $\mathcal{D}$ returns $h(M_c \| r_c) = s^*$. Otherwise, the decoder distinguishes the following two cases:

Case 1: $r \notin R_i$. The decoder $\mathcal{D}$ generates a random syndrome s from $\mathbb{F}_2^{n-k}$ and returns the value $h(M_i \| r) = s$.

Case 2: $r \in R_i$. The decoder $\mathcal{D}$ generates a random vector x from $\mathbb{F}_2^n$ such that $\mathrm{wt}(x) \le t$, and returns $h(M_i \| r) = Hx^T$.

The decoder has to consider the problem that the c-th hash query made by the forger comes in the form $M_c \| r_c$, where $r_c \in R_c$, which means that $h(M_c \| r_c)$ has to be defined as a seemingly decodeable syndrome. If this event occurs, the decoder aborts and admits failure. For convenience, we denote this event by E_{abort}. Also note that $\Pr[E_{abort}] \le \frac{q_{sig}}{2^{n-k}}$.

When the forger makes a signature query for M_i, the decoder picks a random r from the list R_i and discards it from the list. Since there are at most q_{sig} signature queries and the list R_i initially contains q_{sig} elements, this is always possible. Then the decoder $\mathcal{D}$ distinguishes the following two cases:

Case 1: If there has already been a hash query for $M_i \| r$, we have $h(M_i \| r) = s = Hx^T$ for some randomly chosen x. The decoder returns x as the signature together with s and r. The forger can simply check the validity of the signature by deciding whether the equality $Hx^T = s$ holds or not.

Case 2: If $M_i \| r$ has not been queried before, the decoder generates a random vector x from $\mathbb{F}_2^n$ such that $\mathrm{wt}(x) \le t$, and returns x as the signature along with $h(M_i \| r) = Hx^T = s$ and r. The forger can simply verify the validity of the signature by checking whether the equality $Hx^T = s$ holds or not.

Now, we argue that the view of the forger is indistinguishable in the two cases, whether she is facing a real signature challenger or the decoder. First, for the hash oracle, we ignore the event E_{abort} since from the above discussion the probability that this event happens is negligible. In the real signature case, the distribution of the hash oracle is uniform over the range of the random oracle $h(\cdot)$. Now, in the decoder situation, there are two cases for a hash oracle $h(m \| r)$:

(1) If the random salt $r \notin R_i$, the output has the same distribution as a true random oracle.

(2) If the random salt $r \in R_i$, the output of the hash oracle is $h(m \| r) = Hx^T$, which looks like a decodable syndrome.

The probability that the second case occurs is negligible, which can be bounded by

$$1 - \frac{\binom{N_{total} - q_{sig}}{q_{hash}}}{\binom{N_{total}}{q_{hash}}}.$$

To verify this claim, consider a game as follows: there are in total N_{total} balls and N_{dec} of them are red, the hash queries randomly choose q_{hash} balls without replacement, while the signature queries randomly choose q_{sig} red balls with replacement. The probability that the q_{sig} red balls chosen by the signature queries do not contain any balls chosen by a hash query is lower bounded by the probability that the hash queries only choose the balls, which are not chosen by a signature query (assuming that the signature queries choose q_{sig} different red balls), which is $\binom{N_{total} - q_{sig}}{q_{hash}} / \binom{N_{total}}{q_{hash}}$. With $N_{total} = 2^{n-k} \gg \max(q_{sig}, q_{hash})$, the probability that case (2) occurs is negligible since

$$1 - \binom{N_{total} - q_{sig}}{q_{hash}} \Big/ \binom{N_{total}}{q_{hash}} = 1 - \frac{\Theta((2^{n-k} - q_{sig})^{q_{hash}})}{\Theta((2^{n-k})^{q_{hash}})} = O(2^{k-n}).$$

Hence, it is evident that the case (2) occurs with negligible probability and the view of the forger is almost identical between the cases, when she is facing a signature challenger or a decoder of case (1). Combining the arguments above, we claim that the view of the forger is indistinguishable between the cases, when she is facing a signature challenger or a decoder.

When the forger outputs a forgery (M, r, x), we assume that it has already made a hash query for $(M||r)$ but none of the signature queries for M has used the salt r, so $M = M_i$ and $r = r_i$ for some i. Otherwise, the decoder goes ahead and simulates a hash query for $(M||r)$. We justify the assumption that the forger has made a hash query for her forgery $(M||r)$ as follows. If the forger has not made such a hash query before, the decoder simulates the query for $(M||r)$ and we denote the result as $s = h(M||r)$. First, we show that none of the signature queries for M has used the salt r. If it is an EUF-CMA forgery, which means that no signature of the message M has been queried before, then certainly $(M||r)$ has not been used in any of the signature queries. If it is an SEUF-CMA forgery, the forger may have queried the signature oracle for the message M. However, none of the signature oracle queries for M uses r as salt. Otherwise, (M, r, x) does not make a valid forgery, since it must be some signature, which has already been queried by the forger. Next, we show that M must be equal to some M_i for $i \in [1, \dots, q_{hash}]$ with overwhelming probability. If not, then the probability that $h(M||r) = s$ is at most 2^{k-n}. Therefore, with probability at least $1 - 2^{k-n}$, $(M||r) = (M_i||r_i)$ for some i.

Let S_F denote the event that the forger $\mathcal{F}$ succeeds in the experiment with the constraint that it must have made a hash query on the forgery message and random salt (i.e., it has queried the hash of $(M||r)$ as we justified above). Note that if $\mathcal{F}$ succeeds it means that $\mathcal{F}$ outputs a valid forgery (M, r, x), which satisfies $M = M_i$ and $r = r_i$ for some i. If $i = c$, then we get $H_0 x^T = h(M_c||r_c) = s^*$, since the c-th hash query was answered by returning s^*. So the decoder can solve the instance of the PGSDP by returning $e = x$ as the answer. The forger $\mathcal{F}$ succeeds in this experiment if and only if the decoder does not abort at the c-th query, $\mathcal{F}$ has made a hash query for $M||r$, and it outputs a valid forgery for message M with the salt r. Hence,

$$\Pr[S_F] \geq (1 - \frac{q_{sig}}{2^{n-k}})(1 - 2^{k-n})\epsilon.$$

Recall that ϵ' denotes the probability that the decoder successfully solves its decoding instance. Clearly,

$$\epsilon' = \Pr[S_F \wedge i = c] \geq \frac{1}{q_{hash}} \Pr[S_F] \geq \frac{1}{q_{hash}} \left(1 - \frac{q_{sig}}{2^{n-k}}\right)(1 - 2^{k-n})\epsilon,$$

which translates into

$$\epsilon \leq \frac{2^{n-k}}{2^{n-k} - q_{sig}} \frac{1}{1 - 2^{k-n}} q_{hash} \epsilon'.$$

Now, let us consider the time, which it takes for the decoder to solve the PGSDP. In the experiment, the decoder generates at most $q_{sig}(q_{hash} + q_{sig})$ random elements as salts. When it answers a hash query, it searches a list of length q_{sig} and when it answers a signature query, it computes a syndrome for an (n, k) Goppa code. So, we have

$$t' = t + q_{sig}(q_{hash} + q_{sig})\Theta(n - k) + q_{hash}\Theta(n - k) + q_{sig} T_s(n, k),$$

which translates into

$$t = t' - q_{sig}(q_{hash} + q_{sig})\Theta(n - k) - q_{hash}\Theta(n - k) - q_{sig} T_s(n, k),$$

where $T_s(n, k)$ denotes the time to compute a syndrome for an (n, k) Goppa code. □

4 Security against Key Substitution Attacks

Given a signature scheme $\Sigma = (\mathbf{Gen}, \mathbf{Sign}, \mathbf{Verify})$, a key substitution attack (with malicious signer) is a probabilistic polynomial time algorithm KSA which on input of valid parameters, outputs two different valid public keys pk and pk' and a message/signature pair (M, σ), where the verification algorithm accepts on input both (M, σ, pk) and (M, σ, pk').

A key substitution attack KSA is called *weak* if the KSA also needs to output private keys corresponding to public keys, otherwise the KSA is called *strong*. The proof of security against strong key substitution attacks for $mCFS$ is shown in Theorem 4.1.

Theorem 4.1. *The modified CFS scheme mCFS is secure against strong key substitution attacks.*

Proof. Suppose $\mathcal{A}_{KSA}$ can successfully attack mCFS with non-negligible probability. Let $\mathcal{B}_{CRH}$ be an adversary who wants to find a collision of h. Now, as the signer is malicious, the signer can provide sk to $\mathcal{A}_{KSA}$ after signing. Still, to succeed, $\mathcal{A}_{KSA}$ has to find some pk' such that $h(M||pk||r_i) = h(M||pk'||r_i)$. It is required because $\mathcal{A}_{KSA}$ has to pass the verification $Hx^T = H'x^T$. Now, after getting success, $\mathcal{A}_{KSA}$ hands over $(M||pk||r_i, M||pk'||r_i)$ to $\mathcal{B}_{CRH}$. Definitely it is a collision for h. So, the probability of getting a collision of h is at least as large as the success probability of $\mathcal{A}_{KSA}$. But, h is assumed to be collision resistant. $\mathcal{A}_{KSA}$ cannot get success with non-negligible probability, and the claim follows. □

5 Conclusion

We showed that the code-based mCFS signature is both SEUF-CMA secure and KSA secure in the random oracle model under hardness of the Permuted Goppa Syndrome Decoding Problem. It remains an interesting challenge to construct (efficient) code-based signatures, which are provably secure under weaker assumptions and/or provably secure in the standard model.

Acknowledgement: The authors would like to thank Fang Song for his helpful comments. The authors would also like to thank the anonymous reviewers of this Special Issue for their valuable comments that helped us to improve the quality of this manuscript.

Kirill Morozov was supported in part by a Kakenhi Grant-in-Aid for Scientific Research (C) 15K00186 from Japan Society for the Promotion of Science.

References

[1] Moody, D., Post-Quantum Cryptography: NIST's Plan for the Future, Announcement at 7th International Conference on Post-Quantum Cryptography (PQCrypto 2016), https://pqcrypto2016.jp/data/pqc2016_nist_announcement.pdf, February 24, 2016
[2] J. Katz, Digital Signatures, Springer, 2010
[3] Cayrel, P.-L., Meziani, M., Post-quantum Cryptography: Code-Based Signatures, AST/UCMA/ISA/ACN, 2010, 82-99
[4] Phesso A., Tillich J.-P., An Efficient Attack on a Code-Based Signature Scheme, PQCrypto, 2016, 86-103
[5] Courtois, N., Finiasz, M., Sendrier, N., How to Achieve a McEliece-Based Digital Signature Scheme, ASIACRYPT, 2001, 157-174
[6] Alaoui, S. M. E. Y., Cayrel, P.-L., Bansarkhani, R. El, Hoffmann G., Code-Based Identification and Signature Schemes in Software, CD-ARES Workshops, 2013, 122-136
[7] Dallot, L., Towards a Concrete Security Proof of Courtois, Finiasz and Sendrier Signature Scheme, WEWoRC, 2007, 65-77
[8] Faugére, J.-C., Gauthier-Umana, A., Otmani, V., Perret, L., Tillich, J.-P., A Distinguisher for High Rate McEliece Cryptosystems, Information Theory Workshop (ITW), 2011, 282-286
[9] Niederreiter, H., Knapsack-type Cryptosystems and Algebraic Coding Theory, Prob. of Control and Inf. Theory, 1986, 15(2), 159-166
[10] Engelbert, D., Overbeck R., Schmidt, A., A Summary of McEliece-Type Cryptosystems and their Security, Journal of Mathematical Cryptology, 2007, 1, 151-199
[11] Coron, J.-S., Optimal Security Proofs for PSS and Other Signature Schemes, EUROCRYPT, 2002, 272-287
[12] Dou, B., Chen, C.-H., Zhang, H., Key Substitution Attacks on the CFS Signature, IEICE Transactions, 2012, 95-A(1), 414-416
[13] Menezes, A., Smart, N.P., Security of Signature Schemes in a Multi-User Setting, Des. Codes Cryptography, 2014, 33(3), 261-274
[14] Bohli, J.-M., Röhrich, S., Steinwandt, R., Key substitution attacks revisited: Taking into account malicious signers, Int. J. Inf. Sec., 2006, 5(1), 30-36
[15] Mathew, K. P., Vasant, S., Rangan, C. P., A Provably Secure Signature and Signcryption Scheme Using the Hardness Assumptions in Coding Theory, ICISC, 2013, 342-362