A Code-Based Signature Scheme in the Standard Model

O. Blazy[1], P. Gaborit[1], D.T. Mac[1], A. Otmani[2], and J.-P. Tillich[3]

[1] XLIM – Université de Limoges
[2] LITIS – University of Rouen Normandie
[3] SECRET Project – Inria

Abstract. In this work, we propose a code-based signature scheme based on the signature scheme of Kabatianskii, Krouk, and Smeets (known as the KKS scheme), and provide some parameters for it to resist the current attacks. We start from the KKS scheme to construct a chameleon hash function, and from it, build a chameleon signature scheme which is one-time two-tier secure. We also derive a binary tree-based signature scheme from the constructed scheme. The security of our scheme relies on the security of the KKS scheme and the hardness of some code-based problems. We also emphasize that the security of the scheme is considered in the *standard model*.

Keywords: code-based signature · chameleon hash · KKS assumption

1 Introduction

In 1997, Kabatianskii, Krouk and Smeets proposed in [8] a signature scheme based on the difficulty of decoding an $[N, K]$ binary random code. The public key of the scheme consists in a parity-check matrix of this code together with k syndromes of errors whose supports are all included in a small support of size n. The corresponding errors form the secret key of the scheme. They allow to sign a binary message of length k by taking the corresponding linear combinations of these errors. This linear combination is typically an error of rather small weight (since it has weight $\approx \frac{n}{2}$) whose syndrome can be computed by a verifier from the k public syndromes. Later on, several variants of this scheme were proposed [9,3].

However, in 2011, Otmani and Tillich [11] devised attacks against these schemes. They showed that if $\frac{k}{n}$ is not significantly smaller than $\frac{K}{N}$, there is an efficient attack on these schemes. This did not undermine the security of the whole scheme since the attack is still exponential in nature but just showed that the parameters of the scheme have to be chosen carefully.

In the other direction of research, starting with the results of Krawczyk and Rabin [10], the work of Bellare and Shoup [5], and Blazy *et al.* [6], another method of constructing and a new notion of security of signature scheme are proposed, i.e., *chameleon signature* and *two-tier* security. Briefly speaking, a chameleon signature scheme consists of two ingredients: a chameleon hash function, and a regular signature scheme. In this type of signature scheme, the power of the recipient (i.e., possessing the trapdoor of the chameleon hash function) gives the scheme extra properties such as *non-transferability* and *non-repudiation*. [6], this power is given to the signer to strengthen the security, that is, their scheme achieves two-tier security in the standard model.

In this work, we combine the two directions to (i) construct a chameleon hash function from the KKS assumption, and (ii) devise a code-based chameleon-hash signature scheme using this function and also derive a corresponding binary tree-based scheme by using methods of [6]. This gives the first code-based signature scheme with a security proof in the *standard model*. It is also worthwhile

to recall that obtaining an efficient and provably secure scheme in the much weaker random oracle model is already quite a formidable challenge as illustrated by the fact that all the recent code-based signature schemes to the NIST competition for standardizing post-quantum public key cryptography were broken. The signature we propose here is also a post-quantum candidate and the only other candidates for being secure against a quantum computer in the standard model are lattice based signature schemes using bonsai trees and variations of this approach.

The rest of the paper is organized as follows. In Section 2, we recall basic facts on signature schemes; in Section 3, we construct a chameleon hash function whose security is based on the KKS scheme and other hard problems from coding theory; in Section 4, we derive a signature scheme using the constructed chameleon hash function using the technique in [6]; and in the two last sections, we give some concrete parameters for the scheme and draw some conclusions.

2 Preliminaries

2.1 Notation

Vectors are in row form and denoted by bold low-case letters whereas matrices are denoted by bold capital letters. For a given vector $\mathbf{v}$, and a subset J of indices, we let $\mathbf{v}_J = (v_j)_{j\in J}$, the Hamming norm (weight) of $\mathbf{v}$ is denoted by $\|\mathbf{v}\|$; its transpose is denoted by $\mathbf{v}^T$; a similar notation is used for the transpose of a matrix. $x \leftarrow X$ means that x is drawn according to the distribution X, if X is a distribution; or drawn uniformly at random from X when X is a set; or the output of algorithm X, if X is an algorithm.

2.2 Signatures

We recall here the definition of a digital signature scheme.

Definition 1 (Signature scheme). *A digital signature scheme* Sig *with message space* $\mathcal{M}$ *is a triple of probabilistic polynomial-time algorithms,* $\mathsf{Sig} = (\mathsf{Gen}, \mathsf{Sign}, \mathsf{Verify})$*, that verifies:*

- *On input* 1^λ*, algorithm* Gen *outputs a signing key* sk *and a verification key* pk*.*
- *On input a signing key* sk *and a message* $m \in \mathcal{M}$*, algorithm* Sign *outputs a signature* σ*.*
- *On input consisting of a public key and a message-signature pair* (m, σ)*, algorithm* Verify *outputs* 1 (accept) *or* 0 (reject)*.*

Sig *is correct if for any* $\lambda \in \mathbb{N}$*, all* $(\mathsf{pk}, \mathsf{sk}) \leftarrow \mathsf{Gen}(1^\lambda)$*, all* $m \in \mathcal{M}$*, and all* $\sigma \leftarrow \mathsf{Sign}(\mathsf{sk}, m)$*, it holds that* $\mathsf{Verify}(\mathsf{pk}, m, \sigma) = 1$*.*

For the security of a signature scheme, we consider the notion of existentially unforgeability.

Definition 2. *A signature scheme,* Sig*, is* (t, ε, q)*-existential unforgeable under non-adaptive chosen-message attacks* ($\mathsf{EUF\text{-}NCMA}$) *if*

$$\Pr\left[\mathsf{Exp}^{\mathsf{EUF\text{-}NCMA}}_{\mathsf{Sig},\mathcal{F},q}(\lambda) = 1\right] \leq \varepsilon$$

holds for any probabilistic polynomial-time adversary $\mathcal{F}$ *with running time* t *and* q *signature queries, where* $\mathsf{Exp}^{\mathsf{EUF\text{-}NCMA}}_{\mathsf{Sig},\mathcal{F},q}(\lambda)$ *is defined in Table 1. Existential unforgeability under chosen-message attacks is defined similarly.*

Table 1. EUF-NCMA and EUF-CMA experiments for the signature scheme.

Experiment $\mathsf{Exp}^{\mathsf{EUF\text{-}NCMA}}_{\mathsf{Sig},\mathcal{F},q}(\lambda)$:	**Experiment** $\mathsf{Exp}^{\mathsf{EUF\text{-}CMA}}_{\mathsf{Sig},\mathcal{F},q}(\lambda)$:
$\mathcal{Q} := (m_1, \ldots, m_q) \leftarrow \mathcal{F}(1^\lambda)$;	$(\mathsf{pk}, \mathsf{sk}) \leftarrow \mathsf{Gen}(1^\lambda)$;
$(\mathsf{pk}, \mathsf{sk}) \leftarrow \mathsf{Gen}(\lambda)$;	$(m^*, \sigma^*) \leftarrow \mathcal{F}^{\mathtt{OSign}(\cdot)}(\mathsf{pk})$, where the oracle
$\sigma_i \leftarrow \mathsf{Sign}(\mathsf{sk}, m_i)$ for $i = 1, \ldots, q$;	$\mathtt{OSign}(\cdot) := \mathsf{Sign}(\mathsf{sk}, \cdot)$
$(m^*, \sigma^*) \leftarrow \mathcal{F}(\mathsf{pk}, \sigma_1, \ldots, \sigma_q)$;	If $\mathsf{Verify}(\mathsf{pk}, m^*, \sigma^*) = 1$ and $m^* \notin \mathcal{Q} := \{m_1, \ldots, m_q\}$,
If $\mathsf{Verify}(\mathsf{pk}, m^*, \sigma^*) = 1$ and $m^* \notin \mathcal{Q}$,	where m_i is the i-th query, then return 1;
then return 1, else return 0.	else return 0.

2.3 Two-Tier Signatures

We recall the notion of two-tier signature schemes due to M. Bellare and S. Shoup [5]. In a two-tier signature scheme, the key generation algorithm is split into two algorithms, the primary and secondary key generation algorithms. The primary key is static and used for all signatures. The secondary key is ephemeral and used for only one signature. The following definition is from [6], which is a generalization of two-tier signature.

Definition 3 (d-time two-tier signature scheme). *A* two-tier *signature scheme,* TTSig*, is a quadruple of probabilistic polynomial-time algorithms,* $\mathsf{TTSig} = (\mathsf{PriGen}, \mathsf{SecGen}, \mathsf{TTSign}, \mathsf{TTVerify})$*, satisfying that:*

- *On input* $1^\lambda, d$*,* PriGen *outputs a primary signing key* psk *and a primary verification key* ppk*.*
- *On input* ppk *and* psk*,* SecGen *outputs a fresh verification and signing key pair* $(\mathsf{spk}, \mathsf{ssk})$*.*
- *On input* $\mathsf{psk}, \mathsf{ssk}$ *and a message* m*, algorithm* TTSign *outputs a signature* σ*. We denote the stateful variant by* $\mathsf{TTSign}(\mathsf{psk}, \mathsf{ssk}, m; j)$*, where* j *is the state.*
- *On input* $\mathsf{ppk}, \mathsf{spk}$*, a message* m *and a signature* σ*, algorithm* $\mathsf{TTVerify}$ *deterministically outputs* 1 (accept) *or* 0 (reject)*. We denote the stateful variant by* $\mathsf{TTVerify}(\mathsf{ppk}, \mathsf{spk}, m, \sigma; j)$*.*

Security is stated in the following definition.

Definition 4 (Security of two-tier signature scheme). *A two-tier signature scheme* TTSig *is* (t, q, d, ε)*-existential unforgeable under non-adaptive chosen-message attacks* (TT-EUF-NCMA) *if*

$$\Pr\left[\mathsf{Exp}^{\mathsf{TT\text{-}EUF\text{-}NCMA}}_{\mathsf{TTSign},\mathcal{F},q}(\lambda, d) = 1\right] \leq \varepsilon$$

holds for any probabilistic polynomial-time adversary $\mathcal{F}$ *with running time* t*, where* $\mathsf{Exp}^{\mathsf{TT\text{-}EUF\text{-}NCMA}}_{\mathsf{TTSign},\mathcal{F},q}(\lambda, d)$ *is defined in Table 2. Existential unforgeability under adaptive chosen-message attacks* (TT-EUF-CMA) *is defined similarly.*

2.4 Chameleon Hash Functions

The notion of chameleon hash function was introduced by Krawczyk and Rabin [10]. Here, we briefly recall the definition and some of its properties.

Definition 5. *A* chameleon hash *function is defined as* $\mathsf{CHF} = (\mathsf{CHGen}, \mathsf{CHash}, \mathsf{Coll})$*, where:*

Table 2. TT-EUF-NCMA and TT-EUF-CMA experiments for two-tier signature scheme.

Experiment $\mathsf{Exp}^{\mathsf{TT\text{-}EUF\text{-}NCMA}}_{\mathsf{TTSign},\mathcal{F},q}(\lambda, d)$: $(\mathsf{ppk}, \mathsf{psk}) \leftarrow \mathsf{PriGen}(1^\lambda, d)$; $(m^*, \sigma^*, i^*) \leftarrow \mathcal{F}^{\mathtt{NTTSign}(\cdot)}(\mathsf{ppk})$; If $\mathsf{TTVerify}(\mathsf{ppk}, \mathsf{spk}_{i^*}, m^*, \sigma^*) = 1$ and $m^* \notin \mathcal{Q}_{i^*}$, then return 1, else return 0.	**Experiment** $\mathsf{Exp}^{\mathsf{TT\text{-}EUF\text{-}CMA}}_{\mathsf{TTSign},\mathcal{F},q}(\lambda, d)$: $(\mathsf{ppk}, \mathsf{psk}) \leftarrow \mathsf{PriGen}(1^\lambda, d)$; $(m^*, \sigma^*, i^*) \leftarrow \mathcal{F}^{\mathtt{OSKey}(), \mathtt{TTSign}(\cdot,\cdot)}(\mathsf{ppk})$; If $\mathsf{TTVerify}(\mathsf{ppk}, \mathsf{spk}_{i^*}, m^*, \sigma^*) = 1$ and $m^* \notin \mathcal{Q}_{i^*}$, then return 1, else return 0.
	Oracle $\mathtt{OSKey}()$: $i = i + 1$ and $j_i = 0$; $(\mathsf{spk}_i, \mathsf{ssk}_i) \leftarrow \mathsf{SecGen}(\mathsf{ppk}, \mathsf{psk})$; Return spk_i.
Oracle $\mathtt{NTTSign}(m_1, \ldots, m_d)$: $i = i + 1$ and $(\mathsf{spk}_i, \mathsf{ssk}_i) \leftarrow \mathsf{SecGen}(\mathsf{ppk}, \mathsf{psk})$; $\sigma_j \leftarrow \mathsf{TTSign}(\mathsf{psk}, \mathsf{ssk}_i, m_j)$ for $j = 1, \ldots, d$; Store $(m_1, \ldots, m_d)$ in the list $\mathcal{Q}_i$; Return $(\mathsf{spk}_i, \sigma_1, \ldots, \sigma_d)$.	**Oracle** $\mathtt{TTSign}(i', m)$: $j_{i'} = j_{i'} + 1$; $m_{j_{i'}} := m$ If $j_{i'} > d$ or $(\mathsf{spk}_{i'}, \mathsf{ssk}_{i'})$ is undefined then return $\perp$; $\sigma \leftarrow \mathsf{TTSign}(\mathsf{psk}, \mathsf{ssk}_{i'}, m_{j_{i'}})$ and store $m_{j_{i'}}$ in $\mathcal{Q}_{i'}$; Return σ.

- $\mathsf{CHGen}(1^\lambda)$ *outputs a hash key* chk *and the corresponding trapdoor* td.
- $\mathsf{CHash}(\mathsf{chk}, m, r)$ *outputs the hash value* h.
- $\mathsf{Coll}(\mathsf{td}, (m, r), \widehat{m})$ *outputs a randomness* $\widehat{r}$ *such that* $\mathsf{CHash}(\mathsf{chk}, m, r) = \mathsf{CHash}(\mathsf{chk}, \widehat{m}, \widehat{r})$.

Security of chameleon hash function is stated as follows.

Definition 6. *A chameleon hash function* CHF *is said to be* (t, ε)*-collision resistant if for an adversary* $\mathcal{A}$ *running in time at most* t*, it holds that*

$$\Pr_{\substack{(\mathsf{chk},\mathsf{td}) \leftarrow \mathsf{CHGen}(1^\lambda) \\ ((m_1, r_1) \neq (m_2, r_2)) \leftarrow \mathcal{A}(\mathsf{chk})}} \left[\mathsf{CHash}(\mathsf{chk}, m_1, r_1) = \mathsf{CHash}(\mathsf{chk}, m_2, r_2)\right] \leq \varepsilon.$$

A chameleon hash function $\mathsf{CHash}(\mathsf{chk}, \cdot, \cdot)$ with hash key chk and corresponding trapdoor td has to meet the following properties:

1. **Collision resistance:** There is no efficient algorithm that can find two pairs (m_1, r_1) and (m_2, r_2) with $m_1 \neq m_2$ such that $\mathsf{CHash}(\mathsf{chk}, m_1, r_1) = \mathsf{CHash}(\mathsf{chk}, m_2, r_2)$.
2. **Trapdoor collision:** Given td, there exists an efficient algorithm that on any pair (m_1, r_1) and a message $m_2 \neq m_1$ finds a value r_2 such that $\mathsf{CHash}(\mathsf{chk}, m_1, r_1) = \mathsf{CHash}(\mathsf{chk}, m_2, r_2)$.
3. **Uniformity:** All messages m induce the same probability distribution on $\mathsf{CHash}(\mathsf{chk}, \cdot, \cdot)$ for r chosen randomly. This statement can be relaxed to require that the distributions induced by different messages are *computationally indistinguishable.*

2.5 Difficult Problems

In this section, we state some code-based problems which are believed to be hard. The symbol $\mathbb{F}$ means some finite field.

Definition 7 (Syndrome Decoding distribution-SD). *For positive integers* n, k*, and* w*, the* $\mathsf{SD}(n, k, w)$ *Distribution chooses* $\mathbf{H} \leftarrow \mathbb{F}^{(n-k)\times n}$ *and* $\mathbf{x} \leftarrow \mathbb{F}^n$ *such that* $\|\mathbf{x}\| = w$*, and outputs* $(\mathbf{H}, \mathbf{H} \cdot \mathbf{x}^T)$.

Definition 8 (Decision SD Problem-DSD). *On input* $(\mathbf{H}, \mathbf{y}^T) \leftarrow \mathbb{F}^{(n-k)\times n} \times \mathbb{F}^{n-k}$, *the Decision* SD *Problem,* $\mathsf{DSD}(n, k, w)$, *asks to decide with non-negligible advantage whether* $(\mathbf{H}, \mathbf{y}^T)$ *came from the* $\mathsf{SD}(n, k, w)$ *distribution or the uniform distribution over* $\mathbb{F}^{(n-k)\times n} \times \mathbb{F}^{n-k}$.

These two definitions are from [1], and with the assumption that the $\mathsf{DSD}(n, k, w)$ is hard. We also consider the case when the weight of errors varies in an *acceptable interval*.

Definition 9 (Extended extSD distribution). *For positive integers* n, k, a, b, *with* $a \leq b$, *the* $\mathsf{extSD}(n, k, a, b)$ *Distribution chooses* $\mathbf{H} \leftarrow \mathbb{F}^{(n-k)\times n}$ *and* $\mathbf{x} \leftarrow \mathbb{F}^n$ *such that* $a \leq \|\mathbf{x}\| \leq b$, *and outputs* $(\mathbf{H}, \sigma(\mathbf{x}) = \mathbf{H} \cdot \mathbf{x}^T)$.

Definition 10 (Extended Decision extSD Problem). *On input* $(\mathbf{H}, \mathbf{y}^T) \leftarrow \mathbb{F}^{(n-k)\times n} \times \mathbb{F}^{n-k}$, *the Decision* extSD *Problem,* $\mathsf{extDSD}(n, k, a, b)$, *asks to decide with non-negligible advantage whether* $(\mathbf{H}, \mathbf{y}^T)$ *came from the* $\mathsf{extSD}(n, k, a, b)$ *distribution or the uniform distribution over* $\mathbb{F}^{(n-k)\times n} \times \mathbb{F}^{n-k}$.

Here, we make the assumption that the $\mathsf{extDSD}(n, k, a, b)$ problem is hard. In [8], besides providing that the weight of the error lies in $[t_1, t_2]$, the authors also reveal a matrix $\mathbf{F}$, which is closely related to the matrix $\mathbf{H}$. The problem above just provides the information on the weight of the error and nothing more, and in fact, it can be seen as a corollary of Conjecture 3 in [2].

3 The Transformation

3.1 KKS scheme

The KKS scheme uses two codes: a linear code defined by an $(N - K) \times N$ parity-check matrix $\mathbf{H}$ over $\mathbb{F}_q$; a linear code $\mathcal{C}_{\text{hid}}$ over $\mathbb{F}_q$ of length $n \leq N$ and of dimension k which is defined by a $k \times n$ generator matrix $\mathbf{G}$. The code $\mathcal{C}_{\text{hid}}$ has the property that there exist two positive integers $t_1 \leq t_2$ such that with high probability, $t_1 \leq \|\mathbf{c}\| \leq t_2$ for any non-zero codeword $\mathbf{c} \in \mathcal{C}_{\text{hid}}$. The description of the scheme is as follows.

1. $\mathsf{Gen}(1^\lambda)$: The signer
   - chooses parameters N, K, n, k, t_1, and t_2 with respect to the security parameter λ;
   - draws a random $(N - K) \times N$ matrix $\mathbf{H}$; chooses an n-subset $J \subset \{1, \ldots, N\}$;
   - chooses a random $k \times n$ generator matrix $\mathbf{G}$ that defines a code $\mathcal{C}_{\text{hid}}$ such that with high probability, $t_1 \leq \|\mathbf{c}\| \leq t_2$ for any non-zero codeword $\mathbf{c} \in \mathcal{C}_{\text{hid}}$;
   - defines $\mathbf{F} \stackrel{\text{def}}{=} \mathbf{H}_J \mathbf{G}^T$, where $\mathbf{H}_J$ is the restriction of $\mathbf{H}$ to the columns in J;
   - publishes $\mathbf{H}$ and $\mathbf{F}$ as the public key pk, and keeps J and $\mathbf{G}$ as the secret key sk.
2. $\mathsf{Sign}(\mathsf{sk}, \mathbf{x})$:
   - On input a message $\mathbf{x} \in \mathbb{F}_q^k$, the signer computes $\mathbf{v} = \mathbf{x} \cdot \mathbf{G}$.
   - Next, the signer defines the signature $\mathbf{s} = (s_1, \ldots, s_N)$ as $\mathbf{s}_J = \mathbf{v}$, and $s_i = 0$ for $i \notin J$.

3 $\mathsf{Verify}(\mathsf{pk}, (\mathbf{x}, \mathbf{s}))$: On input a pair $(\mathbf{x}, \mathbf{s}) \in \mathbb{F}_q^k \times \mathbb{F}_q^N$, the verifier checks that $t_1 \leq \|\mathbf{s}\| \leq t_2$, and $\mathbf{H} \cdot \mathbf{s}^T = \mathbf{F} \cdot \mathbf{x}^T$.

As noticed in [8], the code $\mathcal{C}_{\text{hid}}$ can be chosen as a random code. A signature corresponds to a random codeword of $\mathcal{C}_{\text{hid}}$ and it is readily seen that its weight lies in an interval $[t_1, t_2]$ around $n/2$ with high probability.

As explained in the introduction, the original KKS scheme with its proposed parameters (and some other variants) was efficiently attacked by Otmani and Tillich in [11]. However, as already pointed out in [11], this attack is of exponential nature and can be avoided if the parameters are chosen carefully. We refer to Section 5 for such a selection. We will therefore make the following assumption.

Assumption 1 (KKS assumption) *There is some region of parameters such that the above scheme is one-time* EUF-CMA.

For our purpose, it will be helpful to adapt the analysis of the key attack of [11] to the case where the attacker has also a KKS signature (the so-called one-time signature scenario). It will be helpful to introduce the following notions:

(*i*) Let $\mathcal{C}_{\text{sec}}$ be the $[N,k]$ code generated by k words $\mathbf{c}_1,\ldots,\mathbf{c}_k$ whose support is included in J and such that $(\mathbf{c}_i)_J = \mathbf{g}_i$, where $\mathbf{g}_i$ is the i-th row of $\mathbf{G}$. In other words, puncturing $\mathcal{C}_{\text{sec}}$ with respect to the positions that are not in J gives $\mathcal{C}_{\text{hid}}$.

(*ii*) Let $\mathcal{C}$ be the sum of $\mathcal{C}_{\text{sec}}$ and the code of parity-check matrix $\mathbf{H}$. It is a code of length N and dimension $K+k$. $\mathcal{C}$ can be easily constructed by an attacker by adding to the code of parity-check matrix $\mathbf{H}$, k words $\mathbf{c}'_1,\ldots,\mathbf{c}'_k$ that are arbitrary solutions of $\mathbf{H}\cdot\mathbf{c}_i'^T = \mathbf{f}_i^T$, where $\mathbf{f}_i$ is the i-th column of $\mathbf{F}$. This comes from the fact that $\mathbf{H}\cdot\mathbf{c}_i^T = \mathbf{f}_i^T$.

The attacker has also in the one-time signature scenario a word $\mathbf{s}$ of weight in the range $[t_1,t_2]$ whose support is included in J and whose syndrome $\mathbf{H}\cdot\mathbf{s}^T$ is a linear combination of columns of $\mathbf{F}$. Let J' be the support of $\mathbf{s}$.

The attack on KKS scheme suggested in [11] amounts to find codewords of $\mathcal{C}_{\text{sec}}$ that reveal the rest of the secret support J by looking for low weight codewords in $\mathcal{C}'$ that is defined to be the code $\mathcal{C}$ punctured in the positions belonging to J'. Therefore, we look for codewords of weight $\leq n-|J'|$ and hope they belong to $\mathcal{C}'_{\text{sec}}$ that is the code $\mathcal{C}_{\text{sec}}$ punctured in J'. The best algorithms for finding low weight codewords in a code consist in variants of information set decoding that look for codewords of some very small weight p in a punctured code $\mathcal{C}''$ whose support contains an information set of $\mathcal{C}$ and whose dimension is slightly larger than the dimension $k+K$ of the code, say $k+K+\ell$, where ℓ is small. In such a case, any codeword of $\mathcal{C}''$ can be completed in a unique way to form a codeword of $\mathcal{C}$. This can be done in polynomial time by standard linear algebra. These algorithms output a non-negligible fraction of codewords of $\mathcal{C}''$ that are of weight p and check whether the completion of those codewords forms a low weight codeword of $\mathcal{C}$. An upper-bound on the complexity for such an algorithm to output a codeword of $\mathcal{C}_{\text{sec}}$ is given by

Proposition 1. *The complexity for outputting one codeword of $\mathcal{C}_{\text{sec}}$ is upper-bounded by*

$$\min_{p,\ell} \frac{C(K+k,\ell,p)\cdot\binom{N-w}{K+k+\ell}}{\sum_{n'=k}^{n-w}\binom{n-w}{n'}\binom{N-n}{K+k-\ell-n'}\cdot\min\left(1,2^{k-n'}\binom{n'}{p}\right)},$$

where $w \stackrel{\text{def}}{=} |J'|$ and $C(K+k,\ell,p)$ is the complexity for outputting the codewords of length $k+K+\ell$ and weight p in a code of dimension $k+K$.

3.2 A chameleon hash function

In this section, we construct a chameleon hash function in the relaxed sense using the KKS assumption. First, define two types of sets as

$$\mathcal{S}_d \stackrel{\text{def}}{=} \{\, \mathbf{s} \in \mathbb{F}_q^N \mid \|\mathbf{s}\| = d \,\},$$
$$\mathcal{S}_{[a,b]} \stackrel{\text{def}}{=} \{\, \mathbf{s} \in \mathbb{F}_q^N \mid a \le \|\mathbf{s}\| \le b \,\}.$$

Now, we consider the function $f(\cdot,\cdot,\cdot)$ defined as

$$f(\mathsf{ppk}, \mathbf{x}, \mathbf{s}) \stackrel{\text{def}}{=} \mathbf{F} \cdot \mathbf{x}^T + \mathbf{H} \cdot \mathbf{s}^T,$$

where $\mathsf{ppk} = (\mathbf{F}, \mathbf{H})$ comes from a KKS signature scheme, $\mathbf{x} \in \mathbb{F}_q^k$ random, and $\mathbf{s} \in \mathcal{S}_t$, where $t \in \mathbb{N}$ is defined later. On input a pair $(\mathbf{x}_1, \mathbf{s}_1) \in \mathbb{F}_q^k \times \mathcal{S}_t$ and a message $\mathbf{x}_2 \neq \mathbf{x}_1 \in \mathbb{F}_q^k$, the one who possesses $J, \mathbf{G}$ (i.e., the *trapdoor*) can find an $\mathbf{s}_2 \in \mathcal{S}_{[t-t_1, t_2+t]}$, with assumption that $t \ge t_1$, such that $f(\mathsf{ppk}, \mathbf{x}_1, \mathbf{s}_1) = f(\mathsf{ppk}, \mathbf{x}_2, \mathbf{s}_2)$ as follows:

1. Compute $\mathbf{v}^T = \mathbf{x}_1^T - \mathbf{x}_2^T \in \mathbb{F}_q^k$.
2. Solve the equation $\mathbf{F} \cdot \mathbf{v}^T = \mathbf{H} \cdot \mathbf{s}^T$ for $\|\mathbf{s}\| \in \mathcal{S}_{[t_1,t_2]}$ (using the signing process of the KKS scheme and set $\mathbf{s}$ to be the signature corresponding to $\mathbf{v}$).
3. Define $\mathbf{s}_2 = \mathbf{s} + \mathbf{s}_1 \in \mathbb{F}_q^N$. It can be seen that $\|\mathbf{s}_2\| \le \|\mathbf{s}\| + \|\mathbf{s}_1\| \le t_2 + t$, and $\|\mathbf{s}_2\| \ge t - t_1$.

It is clear that

$$\mathbf{F} \cdot \mathbf{x}_2^T + \mathbf{H} \cdot \mathbf{s}_2^T = \mathbf{F} \cdot \mathbf{x}_2^T + \mathbf{H} \cdot (\mathbf{s} + \mathbf{s}_1)^T = \mathbf{F} \cdot (-\mathbf{v}^T + \mathbf{x}_1^T) + \mathbf{H} \cdot \mathbf{s}^T + \mathbf{H} \cdot \mathbf{s}_1^T = \mathbf{F} \cdot \mathbf{x}_1^T + \mathbf{H} \cdot \mathbf{s}_1^T.$$

A collision is the two pairs $(\mathbf{x}_1, \mathbf{s}_1) \neq (\mathbf{x}_2, \mathbf{s}_2)$ with $\mathbf{x}_1, \mathbf{x}_2 \in \mathbb{F}_q^k$, $\mathbf{s}_1 \in \mathcal{S}_t$, and $\mathbf{s}_2 \in \mathcal{S}_{[t-t_1, t+t_2]}$. For the uniformity, given $\mathbf{x}_1, \mathbf{x}_2 \in \mathbb{F}_q^k$, the uniform property requires that when $\mathbf{s}_1 \leftarrow \mathcal{S}_t$ and $\mathbf{s}_2 \leftarrow \mathcal{S}_{[t-t_1, t+t_2]}$, the induced probability distributions are computationally indistinguishable, i.e.,

$$\{\, f(\mathsf{ppk}, \mathbf{x}_1, \mathbf{s}_1) \mid \mathbf{s}_1 \leftarrow \mathcal{S}_t \,\} \stackrel{c}{\equiv} \{\, f(\mathsf{ppk}, \mathbf{x}_1, \mathbf{s}_2) \mid \mathbf{s}_2 \leftarrow \mathcal{S}_{[t-t_1, t+t_2]} \,\}.$$

We claim that the function f is a chameleon hash function exactly in this sense.

Proposition 2. *Assume that the* $\mathsf{DSD}(N, K, t)$ *and the* $\mathsf{extDSD}(N, K, t - t_1, t + t_2)$ *problems are hard, and the* KKS *signature scheme for* $t_1, t_2, \mathbf{F}, \mathbf{H}$ *is one-time secure, then* f *is a chameleon hash function.*

Proof (sketch). We need to show that the function f defined as above satisfies the three properties of a chameleon hash function.

Collision resistance: This property is guaranteed by the KKS assumption.

Trapdoor collision: One who has the trapdoor can find a collision as above.

Uniformity: Let

$$D_1 = \{\, f(\mathsf{ppk}, \mathbf{x}_1, \mathbf{s}_1) \mid \mathbf{s}_1 \leftarrow \mathcal{S}_t \,\} \qquad \text{and} \qquad D_2 = \{\, f(\mathsf{ppk}, \mathbf{x}_1, \mathbf{s}_2) \mid \mathbf{s}_2 \leftarrow \mathcal{S}_{[t-t_1, t+t_2]} \,\}.$$

Using hybrid arguments, we can conclude that $D_1 \stackrel{c}{\equiv} D_2$.

□

4 A Signature Scheme using f

In this section, we describe a signature scheme using the function f. We follow the methodology from [6], first, we show how to use it to build a one-time two-tier signature scheme, then in a black box manner we move from the one-time two-tier construction to a non-adaptive signature scheme, and then with an extra use of f, we can obtain a regular signature scheme in the standard model.

4.1 A one-time two-tier Scheme

Our first step is establishing a one-time two tier signature. The descriptions of f, $\mathcal{S}_t$, and $\mathcal{S}_{[t-t_1,t+t_2]}$ are as in Section 3. The message space is $\mathbb{F}_q^k$.

- $\mathsf{PriGen}(1^\lambda)$: Use the setting procedure of the KKS scheme. The primary public key $\mathsf{ppk} = (\mathbf{H}, \mathbf{F})$ and the primary secret key is $\mathsf{psk} = (\mathbf{G}, J)$.
- $\mathsf{SecGen}(\mathsf{ppk}, \mathsf{psk})$: Choose $\widehat{\mathbf{s}} \leftarrow \mathcal{S}_t$, and compute $h = \mathsf{CHash}(\mathsf{ppk}, \widehat{\mathbf{x}}, \widehat{\mathbf{s}})$ for some arbitrary public $\widehat{\mathbf{x}} \in \mathbb{F}_q^k$. The secondary public key is $\mathsf{spk} = h$, and the secondary secret key is $\mathsf{ssk} = \widehat{\mathbf{s}}$.
- $\mathsf{TTSign}(\mathsf{psk}, \mathsf{ssk}, \mathbf{x})$: The signer uses the trapdoor $(\mathbf{G}, J)$ to compute a collision as $\mathbf{s} = \mathsf{Coll}(\mathsf{psk}, \widehat{\mathbf{x}}, \widehat{\mathbf{s}}, \mathbf{x})$. The signature on $\mathbf{x}$ is $\mathbf{s} \in \mathcal{S}_{[t-t_1,t+t_2]}$.
- $\mathsf{TTVerify}(\mathsf{ppk}, \mathsf{spk}, \mathbf{x}, \mathbf{s})$: The verifier checks the condition $\mathsf{CHash}(\mathsf{ppk}, \mathbf{x}, \mathbf{s}) = h$.

The security of the scheme is stated in the following theorem.

Theorem 1. *If f is a (t, ε)-collision resistant chameleon hash function, then for any $q \in \mathbb{N}$, TTSig_f is a $(t', q, 1, \varepsilon')$-$\mathsf{TT\text{-}EUF\text{-}NCMA}$ signature, where $\varepsilon' = \varepsilon$, and $t' = t - O(q)$.*

Proof (sketch). Let $\mathcal{F}$ be a probabilistic polynomial-time adversary that $(t', q, 1, \varepsilon')$-breaks the $\mathsf{TT\text{-}EUF\text{-}NCMA}$ security of TTSig_f. Then we construct an adversary $\mathcal{B}$ that (t, ε)-breaks the collision resistance of f.

Simulation. $\mathcal{B}$ simulates $\mathsf{PriGen}(1^\lambda)$ as follows: it sets $\mathsf{ppk} = \mathsf{chk}$ and returns ppk to $\mathcal{F}$. Now, $\mathcal{B}$ does not have the chameleon hash trapdoor and psk is empty. Upon receiving the ith message $\mathbf{x}_i$ from $\mathcal{F}$, $\mathcal{B}$ simulates $\mathtt{NTTSign}(\mathbf{x}_i)$ as follows: it picks a random $\mathbf{s}_i \leftarrow \mathcal{S}_{[t-t_1,t+t_2]}$ and computes $h_i = \mathsf{CHash}(\mathsf{ppk}, \mathbf{x}_i, \mathbf{s}_i)$; defines the secondary public key $\mathsf{spk}_i = h_i$ and return spk_i and the signature $\mathbf{s}_i$. The simulation is computationally indistinguishable from the real execution.

Extracting collision. Once $\mathcal{F}$ outputs a forgery $(\mathbf{x}^*, \mathbf{s}^*, i^*)$, $\mathcal{B}$ aborts if spk_{i^*} is undefined. Otherwise, $\mathcal{B}$ checks if $\mathsf{CHash}(\mathsf{ppk}, \mathbf{x}_{i^*}, \mathbf{s}_{i^*}) = \mathsf{spk}_{i^*} = \mathsf{CHash}(\mathsf{ppk}, \mathbf{x}^*, \mathbf{s}^*)$. If that is the case, then $\mathcal{B}$ returns the collision $((\mathbf{x}^*, \mathbf{s}^*), (\mathbf{x}_{i^*}, \mathbf{s}_{i^*}))$. □

4.2 A non-adaptive Signature Scheme

By adapting the generic constructions from [6] for the above one-time two-tier scheme, we immediately obtain a stateful scheme $\mathsf{BinTree}[\mathsf{TTSign}] = (\mathsf{Gen}, \mathsf{Sign}, \mathsf{Verify})$ using a binary tree of height ℓ where we assume the message space to be of size 2^ℓ.

The signer will implicitly hold a binary tree of depth ℓ. Every node $v \in \{0,1\}^{\leq \ell}$ has a label L_v which is a secondary public key of the two-tier scheme. All nodes can be computed "on the fly." Each leaf is used to sign a single message. When signing message m, the signer takes the leftmost unused leaf $v_\ell \in \{0,1\}^\ell$ in the tree and generates the label $L_{v_\ell} \leftarrow \mathsf{SecGen}(\mathsf{ppk}, \mathsf{psk})$. Define

$L_{v_{\ell+1}} = m$. Then the path from the root v_0 to v_ℓ is computed. For each undefined node v_i on the path, the signer assigns label $L_{v_i} \leftarrow \mathsf{SecGen}(\mathsf{ppk}, \mathsf{psk})$. After that, every node on the path is signed using the label of its parent.

The signer holds a Merkle tree. When signing the nodes on the path, the signer takes the node v_i in the top-down manner and signs both children of v_i under L_{v_i}, $\sigma_{i+1} \leftarrow \mathsf{Sign}(\mathsf{psk}, \mathsf{ssk}_{v_i}, \mathsf{Child}_l || \mathsf{Child}_r)$, where ssk_{v_i} is the secondary secret key associated with node v_i, and Child_l and Child_r are the left and right children of node v_i, respectively. The signer outputs the path and the two-tier signatures on the path as the signature of m.

4.3 Wrapping-up

To obtain a signature scheme, one can now use the classical transformation from [10] using an extra chameleon hash, so another use of the function f on the leaves. In other words, we pick a fresh $\mathbf{s} \leftarrow \mathcal{S}_t$, set $L_{v_{\ell+1}} = f(\mathsf{ppk}, m, \mathbf{s})$, proceed as before to build the tree (using $L_{v_{\ell+1}}$ as the target message), and output both $\mathbf{s}$, and this non-adaptive signature. The resulting signature is stateful. Depending on the applications, one might prefer to move to a stateless scheme, once again, there exist generic techniques, like the one presented by Goldreich in [7], where basically every randomness is generated through a pseudo-random function. However, as here, we need to keep only the ℓ *active* nodes, there is not a huge blow up in memory for the signer, so it is not really worth the trade-off.

5 Parameters

We have chosen the parameters of the scheme such that the best "practical" information set decoding algorithm, namely the BJMM algorithm [4] has complexity at least 2^λ for solving $\mathsf{DSD}(N, K, t)$, $\mathsf{extDSD}(N, K, t-t_1, t+t_2)$ or breaking the KKS assumption. The parameters that we have obtained are given in Table 3. P represents the probability that the signature does not lie in the interval $[t_1, t_2]$.

Table 3. Example of parameters for the one-time two-tier scheme.

λ	N	K	n	k	t_1	t_2	t	P
128	9800	4510	1900	259	844	1056	1046	2^{-20}
192	14700	6765	2850	387	1281	1569	1569	2^{-24}
256	19600	9020	3800	518	1706	2094	2092	2^{-35}

The public key is $\mathsf{pk} = (\mathsf{ppk}, \mathsf{spk}_\varepsilon)$, where $\mathsf{ppk} = (\mathbf{F}, \mathbf{H})$ and $\mathsf{spk}_\varepsilon = \mathsf{SecGen}(\mathsf{ppk}, \mathsf{psk})$. The size of pk is dominated by the size of ppk which is $(N+k)\cdot(N-K)$. If one wants to sign 2^ℓ times, the size of the signature is $\sigma = (v_h, (L_{v_0}, \sigma_0), \ldots, (L_{v_\ell}, \sigma_\ell), (L_{v_{\ell+1}}, \mathbf{s}))$, each L_* is of size about $N-K$, and each $\sigma_*, \mathbf{s}$ is a chameleon hash opening so of size N. Hence the size of σ is $(\ell+1)\cdot\big(2N-K\big) = O(\ell N)$. Table 4 provides some examples of parameters for a security parameter $\lambda = 128$.

6 Conclusion

In this work, we construct a chameleon hash function. From this function, we proceed to construct a one-time two-tier signature scheme, and finally, this scheme allows us to adapt the technique in [6] to build a binary tree-based signature scheme.

Table 4. Examples of parameters for the tree-based scheme.

ℓ	Size of public key $\|\mathsf{pk}\|$ (bytes)	Size of signatures $\|\sigma\|$ (bytes)
8	$2^{22.7}$	16,976
16	$2^{22.7}$	32,066
32	$2^{22.7}$	62,246
64	$2^{22.7}$	122,606

However, the size of the signatures of our scheme is linear in ℓ, that is the number of sub-signatures, this can hardly be considered as practical (the same problem as mentioned in [6]). Another point which is worth mentioning is that the uniform property of the constructed chameleon hash function is in the sense of *computationally indistinguishability* which is weaker than the sense in the original definition.

From the above mentioned shortcomings, two problems are immediately raised, that is, strengthen the uniform property of the chameleon hash function and devise a code-based signature scheme that is practical.

Acknowledgements. This work has been supported in part by the French ANR project CBCRYPT (ANR-17-CE39-0007).

References

1. C. Aguilar, O. Blazy, J.-C. Deneuville, P. Gaborit, and G. Zémor. *Efficient Encryption from Random Quasi-Cyclic Codes.* In *IEEE Transactions on Information Theory*, page 3927–3943, 2018.
2. M. Alekhnovich. *More on average case vs. approximation complexity.* In *Proc. 44th Annual IEEE Symp.* on *Foundations of Computer Science (FOCS)*, pages 298–307, 2003.
3. P. S.L.M. Barreto, R. Misoczki, and M. A. Simplicio Jr. *One-time signature scheme from syndrome decoding over generic error-correcting codes.* In *Journal of Systems and Software*, 84(2), pages 198–204, 2011.
4. A. Becker, A. Joux, A. May, and A. Meurer. *Decoding random binary linear codes in* $2^{n/20}$*: How* $1+1=0$ *improves information set decoding.* In *Advances in Cryptology–EUROCRYPT 2012*, LNCS. Springer, 2012.
5. M. Bellare and S. Shoup. *Two-Tier Signatures, Strongly Unforgeable Signatures, and Fiat-Shamir without Random Oracles.* In T. Okamoto and X. Wang, editors, *PKC 2007*, volume 4450 of *LNCS*, pages 201216. Springer, Apr. 2007.
6. O. Blazy, S. A. Kakvi, E. Kiltz, and J. Pan. *Tightly-Secure Signatures from Chameleon Hash Functions.* In J. Katz, editor, *Public-Key Cryptography–PKC 2015*, volume 9020 of *LNCS*, pages 256279. Springer, 2015.
7. O. Goldreich *Two remarks concerning the Goldwasser-Micali-Rivest signature scheme.* In A.M. Odlyzko, editor, *CRYPTO'86*, volume 263 of *LNCS*, pages 104–110. Springer, Aug. 1986.
8. G. Kabatianskii, E. Krouk, and B. Smeets. *A digital signature scheme based on random error-correcting codes.* In *Proceedings of the 6th IMA International Conference on Cryptography and Coding*, pages 161167, London, UK, 1997. Springer-Verlag.
9. G. Kabatianskii, E. Krouk, and B. Smeets. *Error Correcting Coding and Security for Data Networks: Analysis of the Superchannel Concept.* John Wiley & Sons, 2005.
10. H. Krawczyk and T. Rabin. *Chameleon Signatures.* In *NDSS 2000.* The Internet Society, Feb. 2000.
11. A. Otmani and J.-P. Tillich. *An Efficient Attack on All Concrete* KKS *Proposals.* In *PQCrypto*, volume 7071 of *Lecture Notes in Computer Science*, pages 98116, 2011.